NASA Conference Publication 2463

SIXTH ANNUAL USERS' CONFERENCE

(NASA-CP-2463) SIXTH ANNUAL USERS' CONFERENCE (NASA) 228 p Avail: NTIS HC A11/MF A01 CSCL 09B
N87-23156
--THRU--
N87-23169
Unclas
H1/61 0071751

Proceedings of a conference hosted by Jet Propulsion Laboratory in cooperation with Goddard Space Flight Center held at the Pasadena Conference Center Pasadena, California October 8-10, 1986

NASA

NASA Conference Publication 2463

SIXTH ANNUAL USERS' CONFERENCE

Edited by
Martha Szczur
Goddard Space Flight Center
Greenbelt, Maryland

and

Elfrieda Harris
Science Applications Research
Lanham, Maryland

Proceedings of a conference hosted by
Jet Propulsion Laboratory in cooperation
with Goddard Space Flight Center
held at the Pasadena Conference Center
Pasadena, California
October 8-10, 1986

National Aeronautics
and Space Administration

**Scientific and Technical
Information Branch**

1986

PREFACE

The Jet Propulsion Laboratory in cooperation with Goddard Space Flight Center hosted the Sixth Annual TAE Users'Conference on October 8-10, 1986 at the Pasadena California Conference Center.

The purpose of the conference was to allow users to interchange information on the use of TAE and its various applications as well as facilitate interaction between users and developers on the future of TAE. To meet these intentions, this year's conference offered an excellent balance of time for presentations, demonstrations, special interest discussions and informal discussions.

The presentations and conversations demonstrated that many TAE users have the same needs and goals. There are several TAE sites interested in developing device-independent imaging applications, accessing data bases, designing User Interface Management Systems (UIMS), executing TAE on PCs, creating programming tools and utilizing windowing systems. It is hoped that this meeting will result in solid work towards coordination between TAE sites and future enhancements to TAE.

TAE Project
NASA/Goddard

Acknowledgements:

The Transportable Applications Executive (TAE) is being developed by the NASA/Goddard Space Flight Center and by Century Computing, Inc. The work is sponsored by the NASA Information Systems Office Code (EI) of the Office of Space Science and Applications (Code E) and the Office of Space Tracking and Data Systems (Code T).

Proceedings from the
Sixth Annual TAE User's Conference
October 8 - 10, 1986
Hosted by
Jet Propulsion Laboratory
in cooperation with
Goddard Space Flight Center

TABLE OF CONTENTS

Page

Overview of TAE at JPL

THE VICAR IMAGE PROCESSING EXECUTIVE 3
Dan Stanfill, Jet Propulstion Laboratory

Future Directions of TAE

TAE PLUS: A CONCEPTUAL VIEW OF TAE IN THE SPACE STATION ERA 19
Martha Szczur, NASA/Goddard Space Flight Center

THE EVALUATION AND EXTENSION OF TAE AS A USER INTERFACE MANAGEMENT SYSTEM 31
Brenda Burkhart, Jet Propulsion Laboratory
Ross Sugar, Century Computing, Inc.

TAE WINDOWS 39
Karl Wolf, Century Computing, Inc.

USING TAE TO ACCESS DATABASES 47
Mike Gough, Science Applications Research, Inc.

A DESIGN FOR A NEW CATALOG MANAGER 53
Cheryl Greenhagen, EROS Data Center

NASA REGIONAL PLANETARY IMAGE FACILITY IMAGE RETRIEVAL AND PROCESSING SYSTEM 61
Susan Slavney, Washington University

TAE V1.4 67
Phil Miller, Century Computing, Inc.

New Ports of TAE

PORTING TAE TO THE IBM PC 73
Brian Phillips, Ames Research Center

APPLE MACINTOSH COMPUTER AS A TAE WINDOW MANAGER 79
Richard Gemoets, Appaloosa System

TAE CONVERSION EXPERIENCES FROM VMS TO UNIX ON A SUN WORKSTATION 89
Don Anderson, Arizona State University

Applications Running Under TAE

SOT, A RAPID PROTOTYPE USING TAE WINDOWS 97
Mark Stephens, Dana Miller, NASA/Goddard Space Flight Center
Dave Eike, Carlow Assoc.
Elfrieda Harris, Science Applications Research

2GCHAS–A HIGH PRODUCTIVITY SOFTWARE DEVELOPMENT ENVIRONMENT 111
Larry Babb, Computer Sciences Corporation

TAE Sybsystems

AN OVERVIEW OF THE TAE–CATALOG MANAGER 127
Fred Irani, Science Applications Research

THE DISPLAY OF THE MANAGEMENT SUBSYSTEM VERSION ONE– A USER'S EYE VIEW 153
Dolores Parker, Science Applications Research

Imaging Applications of TAE

RATIONALIZING VICAR WITHIN A TAE FRAMEWORK–SOME PROBLEMS AND SOME SOLUTIONS 169
Colin Harris, Imperial College

POTENTIAL MEDICAL APPLICATIONS OF TAE 179
J. Ben Fahy, Robert Kaucic, Yongmin Kim, University of Washington

NAVAL APPLICATIONS OF A TAE-DERIVED EXECUTIVE 193
Michael Guberek, Stephen Borders, Serge Masse, Global Imaging

AN OVERVIEW OF THE LAND ANALYSIS SYTEM (LAS) 197
Yun-Chi Lu, NASA/Goddard Space Flight Center

DEVELOPMENT OF LAND ANALYSIS DISPLAY MODULES 211
Douglas Gordon, Douglas Hollaren, Laurie Huewe, TGS Technology /EDC

List of Conference Attendees 227

OVERVIEW OF TAE AT JPL

THE VICAR IMAGE PROCESSING EXECUTIVE

Dan Stanfill
Jet Propulsion Laboratory

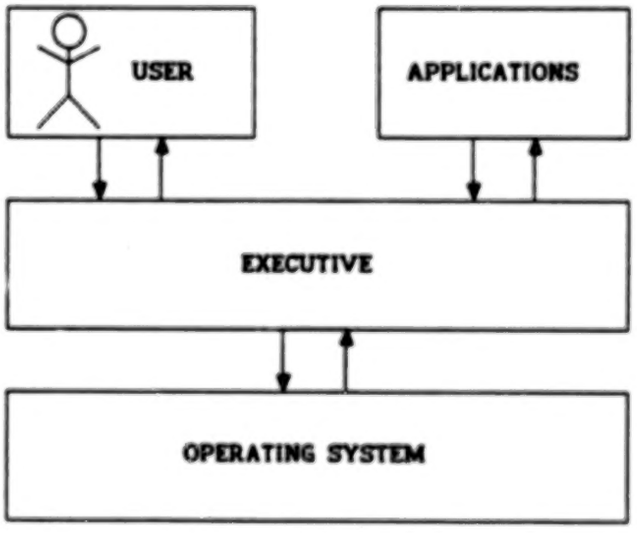
PRECEDING PAGE BLANK NOT FILMED

VICAR

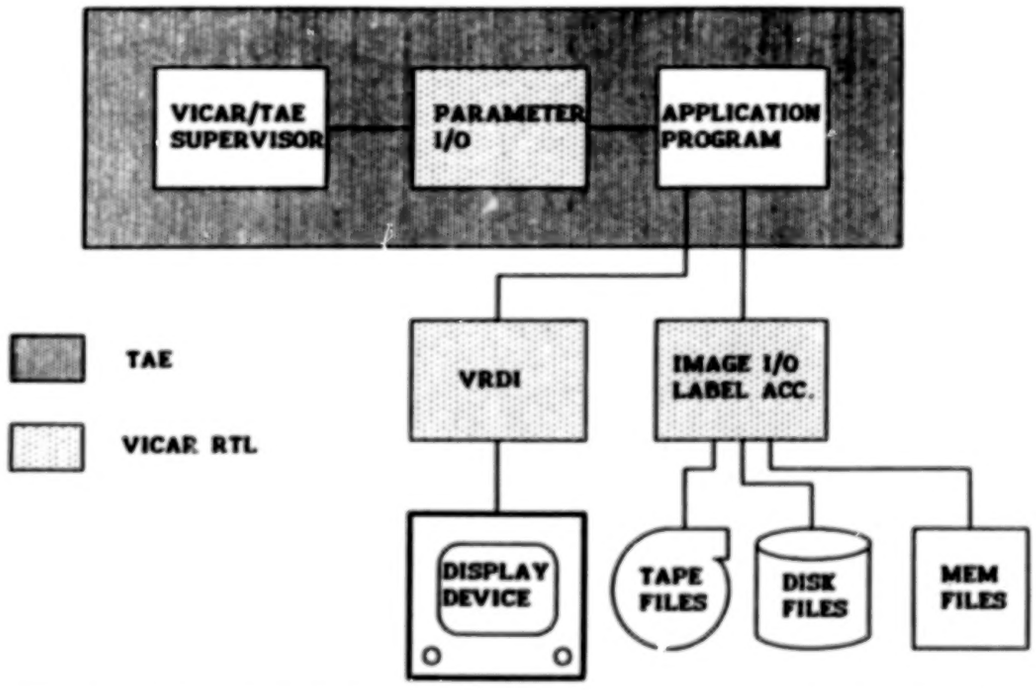
HISTORY

- Completed 1966 on IBM 360/44
- Coded by IBM to JPL design
- Expanded in 1970's to include:
  - → Planetary Imaging
  - → Astronomy
  - → Earth Resources
  - → Land Use
  - → Biomedicine
  - → Forensics
- MIPL started 1982
- Supported SIR-B 1984
- Voyager-Uranus 1986

VICAR EXECUTIVE



VICAR OVERVIEW



VICAR Extensions to TAE

- 10% more code added
- Resource monitor
- Session and batch log changes
- Prompt style interactive parameters
- Magnetic tape handling
- Memory files
- Hardcopy help
- Larger COUNT and string size for TCL
- Batch job name set to job name
- Symbolic $SWITCH options
- EMACS editor access

VICAR
Resource Monitor

- Keeps track of
  - → Proc Start Time
  - → Buffered I/O Count
  - → Direct I/O Count
  - → Page Faults
  - → CPU Time Used
  - → Connect Time
- Available at will through USAGE command
- Generated automatically in batch log
- Optional entry into system data file

VICAR
Batch and Session Logging

- Session Log
  - → Additional log added (SESSION.LOG)
  - → Mirrors interactive session
- Batch Log
  - → Reads like interactive session
  - → Lists proc before executing
  - → Automatic usage statistics on all echoed procs
  - → Traps all VMS output

VICAR

Prompt Style Dynamic Parameters

- Normal TAE provides only Dynamic TUTOR
- Prompt Style is a more terse alternative for advanced users
- User may select either Prompt Style or Dynamic TUTOR
- Prompt Style features
  - → Normal TCL syntax
  - → Just parameters or command plus parameters
  - → Required parameters prompted for if not given
  - → HELP and TUTOR available

VICAR

Prompt Style Example

Problem Description

Want to:

- Choose Prompt Style interactive parameters
- Run the program IMGDISP
- Display the image MIRANDA.DAT
- Zoom by a factor of 2
- Enter TUTOR to look for other features of the program

VICAR

Prompt Style Example

VICAR› LET $DYNAMIC="PROMPT"	Choose Prompt Style
VICAR› IMGDISP	Enter program IMGDISP
-- Enter ESC-ESC for HELP or TUTOR --	
IMGDISP› DISPLAY MIRANDA.DAT	DISPLAY takes 1 parm
IMGDISP› ZOOM	ZOOM requires FACTOR
Enter FACTOR› 2	Forced to give FACTOR
IMGDISP› ZOOM FACTOR=2	Same command again
IMGDISP› ESC-ESC	Enter Escape-Mode
IMGDISP-ESCAPE› TUTOR	Enter TUTOR on IMGDISP

VICAR

Magnetic Tape Handling

- New Intrinsic Commands
  - → ALLOC
  - → DEALLOC
  - → MOUNT
  - → DISMOUNT
  - → REWIND
- Symbolic names for generic procs
- Tape position kept between procs
- Drive information kept between sessions

VICAR

Memory Files

- New intrinsic commands
  - → MAL
  - → DAL
- Invisible to programs
- Used to pipe an image through a series of programs without going to disk
- Decreases I/O requests

VICAR

Other Modifications to TAE

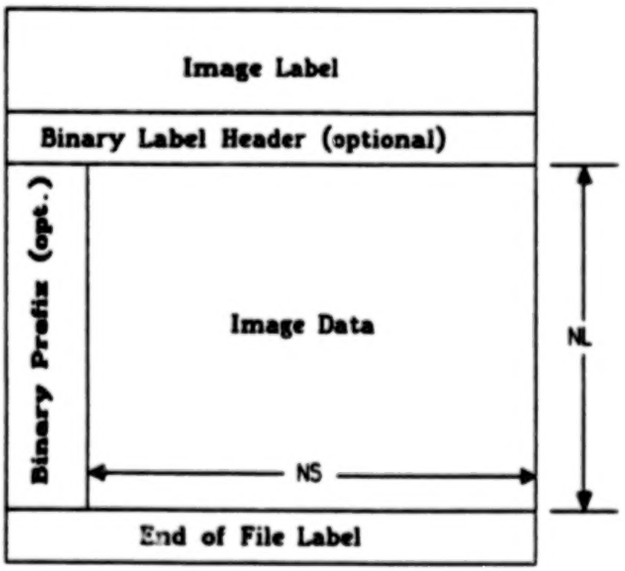
- Hardcopy HELP
  - → HELP-HARDCOPY command added
  - → Reformats HELP and TUTOR information
- TCL limits expanded
  - → Command line up to 4096 characters
  - → COUNT up to 600
  - → String size up to 250 characters
- Process name of batch job set to job name
- $SWITCH options set symbolically via FLAG command
- Access to EMACS editor session

The VICAR Run-Time Library

- Device independent subroutine package
- Designed specifically for the image processing environment
- Equally easy to use for both FORTRAN and C
  - → Strings may be passed by descriptor or reference
- Five inter-related packages:
  - → Image I/O
  - → Image Label Access
  - → Parameter Retrieval
  - → Virtual Raster Display Interface
  - → Utilities

VICAR

Image File Format



VICAR

Image Formats

- BYTE — 8 bit unsigned integer
- HALF — 16 bit signed integer
- FULL — 32 bit signed integer
- REAL — 32 bit floating point
- DOUB — 64 bit floating point
- COMP — Pair of 32 bit floating point

VICAR

Standard Parameters

- INP — Input file name
- OUT — Output file name
- SIZE — Image size
- SL, SS, NL, NS — Image size (alternative)
- S1, S2, S3, S4, N1, N2, N3, N4 — Image size (multi-dimensional)
- FORMAT — Image data format

VICAR

Image I/O Features

- Device independent
  - → Disk
  - → Tape
  - → Memory files
- Fast buffered I/O
- Two I/O methods:
  - → Line oriented I/O
  - → Array I/O
    (mapped memory sections)

VICAR

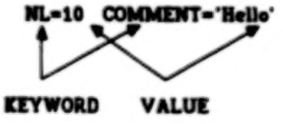
Image I/O Routines

- XVUNIT assigns a unit number to a file to be opened
- XVADD adds information to the internal control block for XVOPEN
- XVOPEN opens an image file
- XVCLOSE closes an image file
- XVGET retrieves internal control block information
- XVREAD reads from an image file
- XVWRIT writes to an image file
- XVSIGNAL signals an error

VICAR

Image Label

- ASCII string composed of "Label Items"
- Composed of "System" and "History" subsets
- Label items are keyword-value pairs
  - → Integer
  - → Floating point
  - → String
- Example:



- Item values retrievable individually

VICAR

Label Access Routines

- XLADD adds a label item
- XLDEL deletes a label item
- XLGET returns the value of a label item
- XLINFO returns information about a given item
- XLHINFO returns information about the history label
- XLNINFO returns the keyword name of the next item in the label
- XLGETLABEL returns the entire label as an ASCII string

VICAR

Parameter I/O Features

- Replaces standard TAE XR subroutines
- Fewer subroutines than standard TAE with more functionality
- Easy to use for both FORTRAN and C
  - → Strings either by descriptor or reference
- Parameter file I/O package
  - → Normal TCL limits can be exceeded
  - → Receiving program doesn't know or care if parms are from a file
  - → Clean communication between programs with parameter verification

VICAR

Parameter I/O Routines

- XVPARM retrieves the value of a parameter
- XVPTST returns TRUE if keyword was specified
- XVPCNT returns the TCL COUNT of a parameter without the value
- XVSPTR helps unpack an array of C strings
- XVINTRACT requests parameters interactively (either through Prompt Style or Dynamic TUTOR)
- XVPOPEN opens a parameter data file for output
- XVPOUT writes a parameter to a data file
- XVPCLOSE closes a parameter data file

VICAR

Virtual Raster Display Interface

- Common set of device independent routines
- Run-time selection of display device
- Devices supported:
  - → Adage 3000 high/low resolution
  - → Deanza IP8500 low resolution
  - → Deanza IP85HR high resolution
  - → Ramtek RM9460 low resolution

VICAR

Utility Routines

- ABEND causes abnormal termination of processing due to error
- XVEND terminates processing
- QPRINT sends output to the user terminal or log
- PRINT sends output to selected devices
- VIC1LAB returns the old style IBM VICAR labels if present
- XVTPMODE returns TRUE if unit is a tape
- XVFILPOS returns current tape position
- XVSFILE positions a tape
- XVSIZE simplifies parsing of SIZE field

VICAR

Current and Future Work

- Multi-dimensional files
  - → Currently being implemented
  - → Up to four dimensions
  - → Dimensions can be "named" for different types of data
  - → May be accessed as 2-dimensional for 2-d programs
- VICAR Interactive Display Manager
  - → Final design stages
  - → Can run applications while using display device
  - → Viewport files
  - → Optimal use of VRDI
- ANSII labeled tapes
- XVMOUNT, XVDISMOUNT
- Automatic image portioning

The VICAR Executive

Summary

- Extended TAE to provide
  - → Resource moniter
  - → Easy to read logs for novice users
  - → Prompt Style for advanced users
  - → Extensions to ease the handling of large volumes of image data
- Run-time Library features:
  - → Fast easy to use I/O— no complicated calling sequences to remember
  - → FORTRAN and C string handling
  - → Complete image label access
  - → Improved parameter handling
  - → VRDI

FUTURE DIRECTIONS OF TAE

N87
23157
UNCLAS

N87-23157

TAE PLUS: A CONCEPTUAL VIEW OF TAE IN THE SPACE STATION ERA

Martha Szczur
NASA/GSFC

PRECEDING PAGE BLANK NOT FILMED

TAE: Current Profile and Future Direction

M. Szczur
NASA/GSFC, Code 521
Data Systems Technology Division
Greenbelt, Md 20771
(301) 286-8609 (FTS) 888-8609

The Transportable Applications Executive (TAE) is a software management system that binds a set of application programs into a single, easily operated system. TAE has packaged a set of common system service functions and user interface functions into a stable framework on which application software can immediately be built. [see Viewgraph 1] TAE was originally developed in the early 1980s to support scientific interactive data analysis applications (e.g., General Meteorology Package (GEMPAK), Atmospheric and Oceanographic Information Processing System (AOIPS), Land Analysis System (LAS), Pilot Climate Data Systems (PCDS)).

In FY86 TAE saw significant growth, in both it's use for new projects and in system development. TAE's user community increased from last year's reported 40 facilities, located at 28 known sites to 110 facilities, located at 65 known sites. [see Viewgraphs 2,3,4] As the use of TAE has grown, the types of applications being built with it has also increased, and now includes scientific analysis systems, image processing, data base management, user assistant/teaching tool, defense systems and prototyping tool.

This last application, using TAE for prototyping user interfaces, has been the prime force behind the new TAE research and development work. The Data Systems Technology Division (Code 500) is developing prototypes of user interfaces for different functions involved in the operation, analysis and data communication of Space Station payloads. [see Viewgraph 5] TAE is a valuable prototyping tool because it enables a developer to build an entire application user interface model and run it without writing a single line of application code. User designers can then directly interact with the "proto" system and can quickly change or configure the system by editing the text files. However, while TAE can be used for prototyping today, there are many enhancements and expansions that

are required when a new user type is introduced -- the user interface designer, who will apply human factor techniques in the development of the applications' interfaces. [see Viewgraph 6]

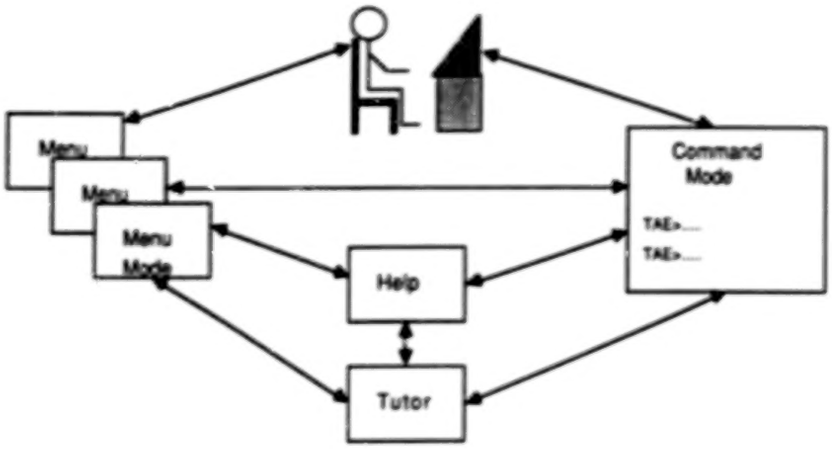
Another force driving new development is the need to update TAE's user interface to support the latest interactive graphic device technology. The current TAE, "TAE_Classic", uses interface techniques designed for an 80x24 character monochrome alphanumeric terminal,and does not effectively utilize features such as windowing, graphics, color, and selection devices available on newer workstations. To meet our needs, development of a "TAE_Plus" began in FY86 and involves augmenting TAE with three different sets of tools:

-- a user interface toolkit for creating generic interface elements,
-- an application toolkit for customizing the generic interface elements for use in a particular application, and
-- run-time service subroutines that will tie the application code to the independently defined interface elements.

The change in structure from "TAE_Classic" to "TAE_Plus" is shown in Viewgraphs 7 and 8.

We are using a phased approach to develop TAE_Plus. [see Viewgraphs 9 and 10] In the first phase, we have meet the needs of our existing community and provided some support for rapid prototyping by developing a TAE "facelift", which adds an enhanced TAE interface (with windowing, mouse interaction, pull-down menus, etc.) to a select set of graphic workstations (SUN 3, VAXStation II/GPX, and Macintosh with MacWorkstation). [see Viewgraphs 11-14] The TAE Facelift allows us to test many new concepts quickly for feedback and performance. In our second phase we will build a fully-integrated user interface management system, TAE_Plus, that supports the separation of interface from application, with the concomitant ability to prototype and rapidly change interfaces. [see Viewgraphs 15,16,17] This robust functionality will support, in an integrated manner, an application's development cycle from the prototype step through to the fully operational system.

SUMMARY OF TAE CLASSIC FEATURES



- UTILITY PROCS
- MESSAGES
- LOGON/LOGOFF PROCS
- SESSION LOG
- PROC & MENU HIERARCHY SEARCH
- GLOBAL VARIABLES
- SAVED PARAMETER SETS
- ASYNC, BATCH, SYNC
- USER COMMANDS
- PROCEDURES
- FLEXIBLE PARAMETER SPECS

TAE USER'S SITES -- 1986

TAE USER'S SITES -- 1986

○ UNIVERSITIES

1. BROWN UNIV.
2. YALE UNIV.
3. CORNELL UNIV.
4. UNIV. OF MARYLAND
5. N.C. STATE UNIV.
6. WASHINGTON UNIV.
7. COLORADO STATE UNIV.
8. UNIV. OF COLORADO
9. NAVAL POST GRAD SCHOOL
10. UNIV. OF CA. BERKELEY
11. UNIV. OF CA. SANTA BARBARA
12. UNIV. OF CA. L.A.
13. UNIV. OF ILLINOIS
14. UNIV. OF HAWAII
15. SCRIPPS INSTITUTE/UCSB
16. IMPERIAL COLLEGE LONDON
17. UNIV. OF NEW MEXICO
18. ARIZONA STATE UNIV. (TUSCON)
19. ARIZONA STATE UNIV. (TEMPE)
20. UNIV. OF MICHIGAN
21. UNIV. OF UTAH (2)
22. UNIV. OF WASHINGTON (2)
23. PURDUE UNIV.
24. UNIV. OF NEBRASKA
25. TEXAS A&M
26. MIT
27. CALTECH
28. NAVAL ACADEMY

□ GOVERNMENT/RESEARCH LABS

1. GSFC (AOIPS/2, LAS, PCDS, PLDS, SEAPAK, DCF,SSTE, OTHERS) (18)
2. E-SYSTEMS
3. ERIM
4. EROS DATA CENTER (3)
5. USGS/FLAGSTAFF, ARIZONA
6. JPL (MIPL, NODS, PLDS, PDS, ASAS, OTHERS?)
7. GLOBAL IMAGING.
8. AMES (ARMY, 2GCHAS, NASA/PLDS)
9. NORTHROP CORP (USAF)
10. SOHIO PETROLEUM CO.
11. LANGLEY RESEARCH CENTER/NASA
12. JOHNSON SPACE CENTER/NASA
13. WRIGHT-PATTERSON AIR FORCE BASE
14. NORDA/NSTL
15. SMITHSONIAN
16. MARSHALL SPACE FLIGHT CENTER/NASA
17. TETRATECH
18. KENNEDY SPACE FLIGHT CENTER/NASA
19. HARRIS CORP.
20. ANALYTICAL METHODS
21. DIA
22. DRAPER LAB (USAF)
23. CENTURY COMPUTING.
24. NAVAL RESEARCH LAB
25. NAVAL OCEANOGRAPHIC FACILITY
26. NATIONAL BUREAU OF STANDARDS
27. EOSAT
28. SASC

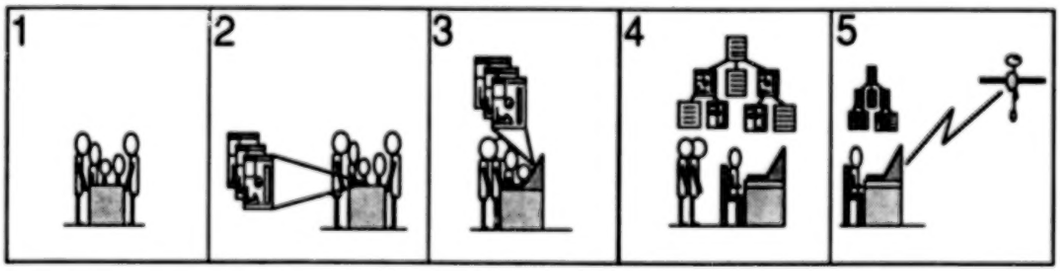
CURRENT TAE INSTALLATIONS

- VAX/VMS
- VAX/UNIX
- SUN/UNIX
- GOULD/UNIX
- APOLLO/UNIX
- HP9000/UNIX
- JUPITER/UNIX

CURRENT TAE PORTING ACTIVITIES

- IBM/PC/XT
- ISI
- CDC 180/840
- PRIME
- IBM mainframe
- CRAY

PROTOTYPING STEPS



STEP 1 Initial Talk Session

STEP 2 Paper Prototype/ Talk-thru Session

STEP 3 Rapid Prototype/ Step-thru Session

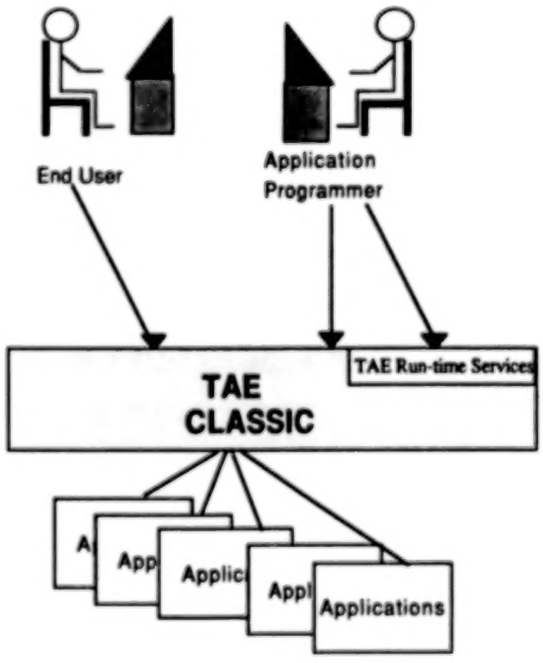
STEP 4 Operational User Interface

STEP 5 Operational Application System

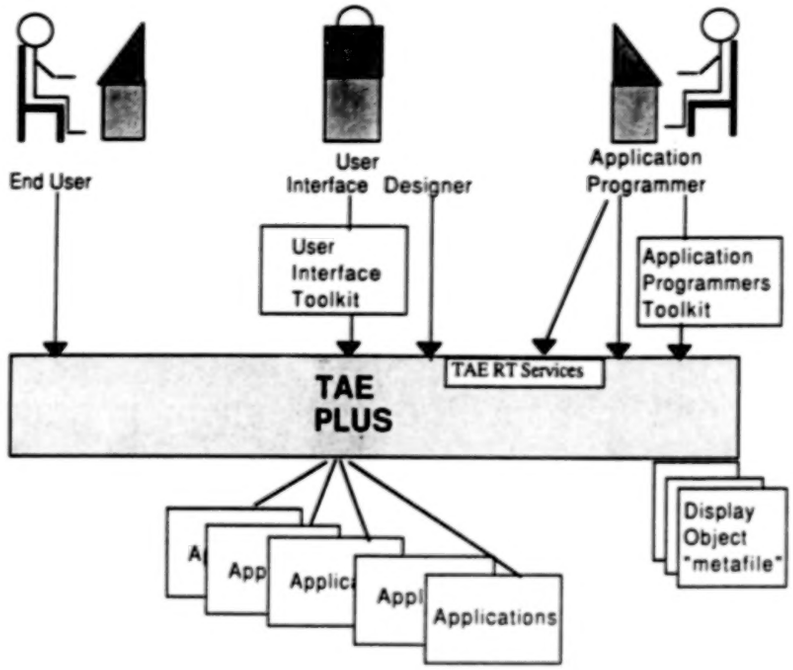
REQUIREMENTS FOR INTEGRATED PROTOTYPING SYSTEM

- take advantage of new hardware technologies
- support prototyping of user interfaces
- separate the user interface from the application
- manage the defined user interface
- unify and manage the application programs
- supply programmer services and tools to easily access all the system's elements

STRUCTURE IN TAE_CLASSIC - BASED SYSTEM



STRUCTURE IN TAE_PLUS - BASED SYSTEM



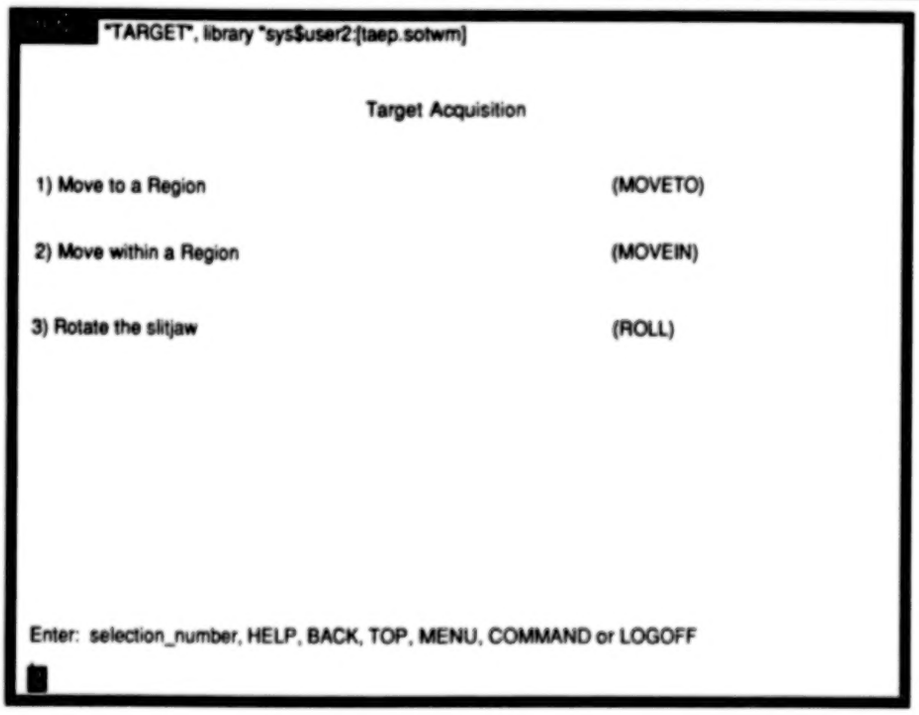
TAE ACTIVITIES AND PLANS

- TAE Facelift
  - SUN3
  - VAXStation II/GPX
  - Macintosh (Macstation/VAX)
- Application Programmer Services
  - MDF/PDF Templates
  - Interface Utility Subroutine Package
  - On-line help for programmers
- Interactive Programmer Toolkit
  - Menu, Tutor, Help Creator/Editor
  - Application Screen Designer

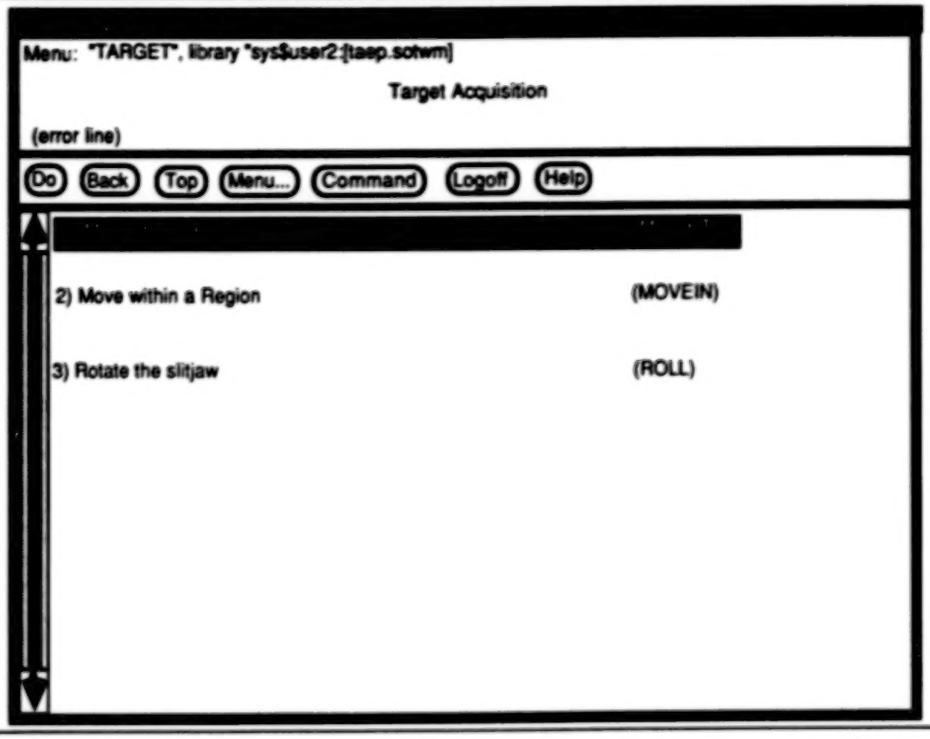
TAE ACTIVITIES AND PLANS

- Interactive User Interface Designer Toolkit
- New Architecture Definition
  - UIMS Model
- RCJM Enhancements
  - TCPIP Protocol
  - Telescience Prototyping

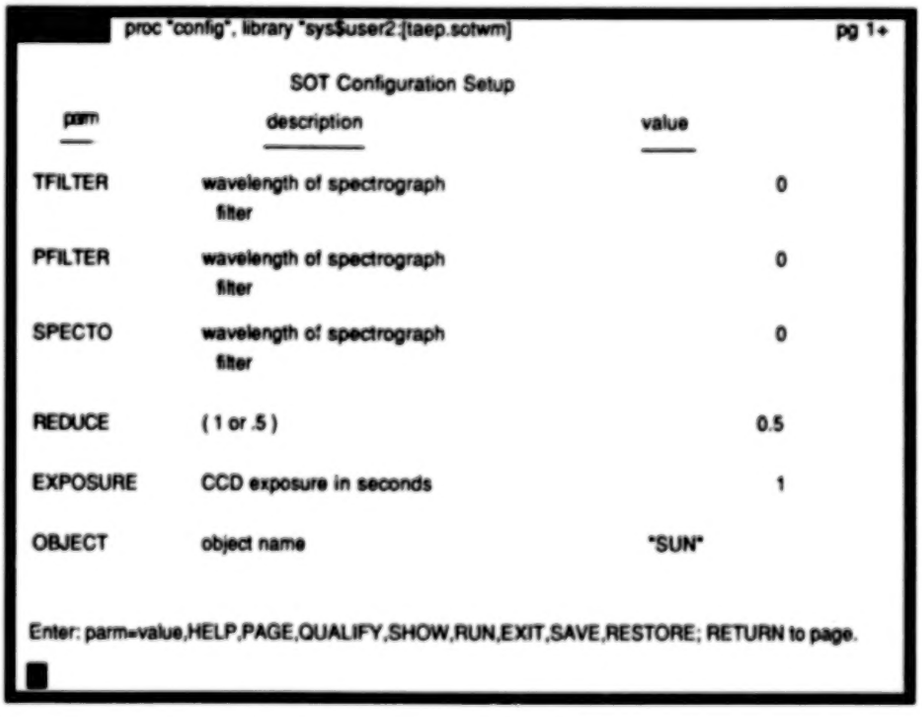
ORIGINAL TAE MENU SCREEN


"TARGET", library "sys$user2:[taep.sotwm]
Target Acquisition
1) Move to a Region (MOVETO)
2) Move within a Region (MOVEIN)
3) Rotate the slitjaw (ROLL)
Enter: selection_number, HELP, BACK, TOP, MENU, COMMAND or LOGOFF

TAE FACELIFT MENU



ORIGINAL TAE TUTOR SCREEN



TAE FACELIFT TUTOR

TUTOR: proc "config", library "sys$user2:[taep.sotwm]

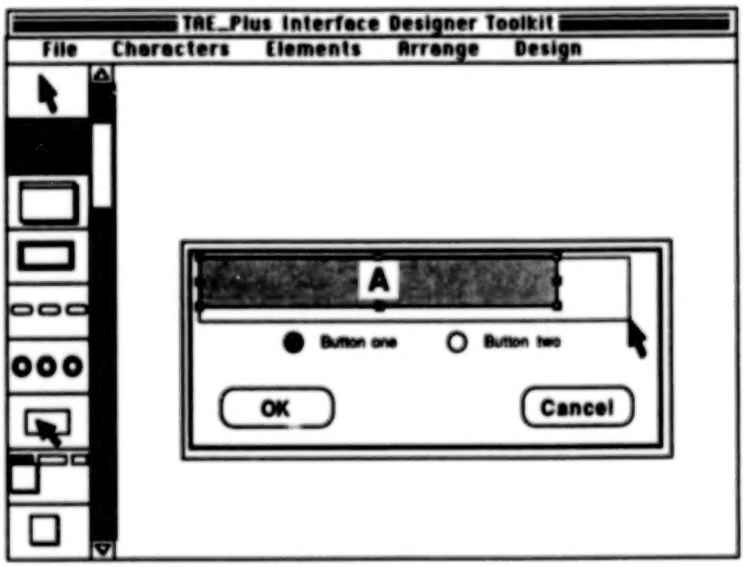
SOT Configuration Setup

(error line)

Run | Show... | Initial | Save... | Restore... | Structure | Exit | Help

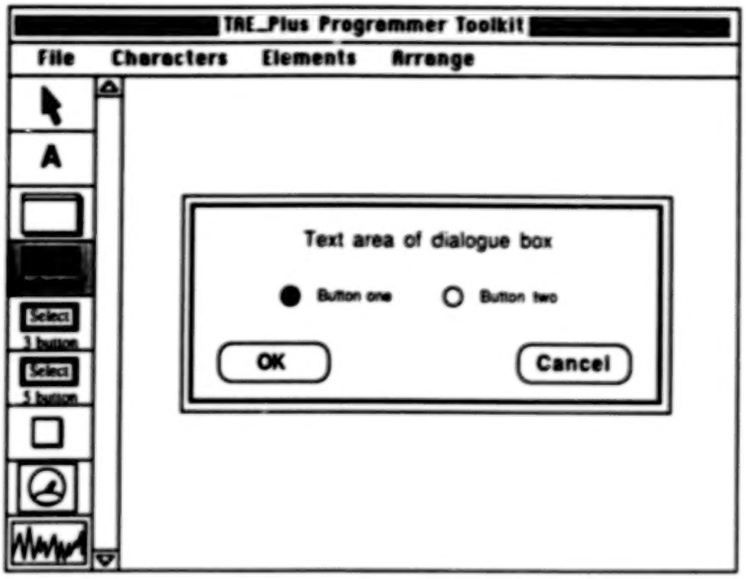
parm	description	value
TFILTER	wavelength of spectrograph filter	0
PFILTER	wavelength of spectrograph filter	0
SPECTO	wavelength of spectrograph filter	0
REDUCE	(1 or .5)	0.5
EXPOSURE	CCD exposure in seconds	1
OBJECT	object name	"SUN"

INTERFACE DESIGNER TOOLKIT SCREEN

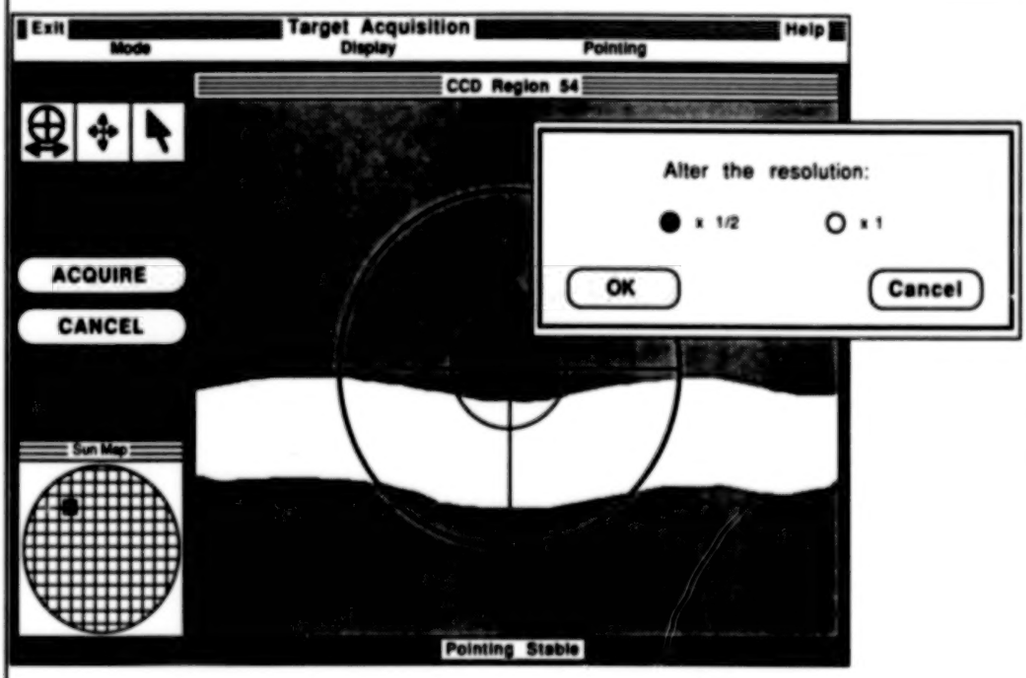


ORIGINAL PAGE IS
OF POOR QUALITY

APPLICATION PROGRAMMER'S TOOLKIT



END-USER'S VIEW OF APPLICATION INTERFACE



N87
23158
UNCLAS

N87-23158

THE EVALUATION AND EXTENSION OF TAE IN THE DEVELOPMENT OF A USER INTERFACE MANAGEMENT SYSTEM

Brenda Burkhart
Jet Propulsion Laboratory

Ross Sugar
Century Computing, Inc.

INTRODUCTION

- LARGE INFORMATION PROCESSING PROGRAM
- JPL
- GRAPHICS AND DISPLAY ENGINEERING GROUP
- USER INTERFACE MANAGEMENT SYSTEM (UIMS)
- TAE

TOPICS

- THE PROJECT
- PROJECT REQUIREMENTS
- UIMS
- REQUIREMENTS SUPPORTED BY TAE
- MODIFICATIONS TO TAE
- LESSONS LEARNED
- EVALUATION OF TAE AS A UIMS

THE PROJECT

- INFORMATION GATHERING AND DISPLAY SYSTEM
- DISTRIBUTED PROCESSING NETWORK
- HOST PROCESSORS (VAX 11/750)
- WORK STATIONS (VAXSTATION II)
  - VAXSTATION DISPLAY + SPECIAL GRAPHICS DISPLAY DEVICE

THE USER

- TRIGGERS VARIOUS USER FUNCTIONS (APPLICATIONS SOFTWARE)
- PERFORMS SOPHISTICATED I/O
  - QUESTION/ANSWER
  - FORM FILLING
  - WORD PROCESSING
  - GRAPHICAL ENTITY MANIPULATION AND DISPLAY

PROJECT REQUIREMENTS

- PROVIDE A CONSISTENT USER INTERFACE TO USER FUNCTIONS
- PROVIDE A CONSISTENT APPLICATIONS INTERFACE TO USER FUNCTIONS
- SUPPORT BOTH COMMAND AND MENU TRIGGERING OF USER FUNCTIONS
- SUPPORT MULTIPLE USER INTERACTION TECHNIQUES
- PROVIDE CONSISTENT HELP AND MESSAGE FACILITIES
- SUPPORT A MULTI-WINDOWED ENVIRONMENT
- SUPPORT A DISTRIBUTED APPLICATIONS ENVIRONMENT

UIMS

"A UIMS IS A SOFTWARE PACKAGE THAT ACTS AS A BUFFER, OR MEDIATOR, BETWEEN APPLICATIONS SOFTWARE AND THE USERS OF A SYSTEM".

H. J. STRUBBE

UIMS CHARACTERISTICS

- CONTROLS USER DIALOGUE
- PROVIDES A CONSISTENT USER INTERFACE
- SUPPORTS A STANDARD APPLICATIONS INTERFACE TO USER FUNCTIONS
- SUPPORTS ONE OR MORE USER FUNCTION TRIGGERS
  - COMMAND
  - MENU
  - ICON

UIMS CHARACTERISTICS (CONTINUED..)

- SUPPORTS SEVERAL USER INTERACTION TECHNIQUES
  - QUESTIONS/ANSWERS
  - FORM FILLING
  - WORD PROCESSING
  - GRAPHICS
  - MENUS
  - MULTIPLE WINDOWS
- SUPPORTS A SET OF INTEGRATED TOOLS TO DEFINE USER FUNCTIONS AND DEVELOP USER DIALOGUE

ADVANTAGES OF UIMS

- USER
  - EASE OF LEARNING SYSTEM
  - PREDICTABILITY
  - GENERAL EASE OF USE
- APPLICATIONS DEVELOPER
  - NO USER INTERFACE SOFTWARE DEVELOPMENT REQUIRED
  - MORE PORTABLE
  - EASIER ON PROGRAMMER
  - USER INTERFACE CAN BE DEVELOPED BY NON-PROGRAMMERS
  - USER INTERFACE CAN CHANGE WITHOUT CHANGE TO APPLICATIONS SOFTWARE

REQUIREMENTS SUPPORTED BY TAE

- SUPPORTED REQUIREMENTS
  - CONSISTENT APPLICATIONS INTERFACE TO USER FUNCTIONS
  - SUPPORT OF BOTH COMMAND AND MENU MODE
  - TOOL PROVIDED FOR DIALOGUE CONTROL (TCL)
- UNSUPPORTED REQUIREMENTS
  - NO SUPPORT FOR PROJECT DISTRIBUTED ENVIRONMENT
  - NO SUPPORT FOR SOPHISTICATED USER INTERFACE

MODIFICATIONS TO TAE

- DISTRIBUTED ENVIRONMENT
  - USED PROJECT COMMUNICATIONS PROTOCOL
- SOPHISTICATED USER INTERFACE
  - INTRODUCTION OF OBJECT DATA TYPE
  - SUPPORT OF SOPHISTICATED TOOLS (FORMS MANAGER, WORD PROCESSOR, GKS, PROJECT SPECIFIC GRAPHICAL OPERATIONS)
  - SUPPORT OF MULTI-WINDOWED ENVIRONMENT

TAE User's Conference
Padadena, California
October 8, 1986

TAE WINDOWS

Karl R. Wolf
Century Computing, Inc.
1100 West Street
Laurel, Maryland 20707
(301) 953-3330

PRECEDING PAGE BLANK NOT FILMED

CENTURY
COMPUTING
DATA SYSTEMS DEVELOPMENT

Topics

- Background
- Original Goals
- Chronology of effort
- A Look into the Future
- Conclusions

CENTURY
COMPUTING
DATA SYSTEMS DEVELOPMENT

Background

- Century is the original developer of TAE.
- TAE Windows began in the spring of 1985.
- To investigate new techniques in user interface technology and apply them to TAE.
- Use the increased capabilities of the new Workstations.

CENTURY
COMPUTING
DATA SYSTEMS DEVELOPMENT

Original Goals

- Must be compatible with existing software.
- Portable
  - User Interface
  - Application Interface
- "Intuitive" and easy to use.
- Multiple windows per process.
- Serial and Concurrent window sharing.
- TCL manipulation of windows.
- Graphics must be supported.
- Live gracefully with the host systems.

CENTURY
COMPUTING
DATA SYSTEMS DEVELOPMENT

Chronology

- VT220 prototype
  - VAX/VMS using the DEC SMG package.
  - Acted as a proof of concept.
- Good points:
  - Inexpensive (VT220s are cheap, and SMG is part of VMS 4.x)
  - Windows may be moved, resized and overlayed.
  - Popup Menus
  - Fast prototyping and excellent documentation
- Bad points:
  - Awkward, no mouse, only cursor keys.
  - Poor resolution and the screen was too small.

CENTURY COMPUTING
DATA SYSTEMS DEVELOPMENT

Chronology (Cont.)

- DEC VAXstation 100
  - First of our Full TAE Windows prototypes
  - A true windowing and bitmapped system
  - Mouse input, popup menus, regions, VT100 emulation
- Good points:
  - Three (3) full window interfaces
    1) Mouse oriented user interface.
    2) Application program interface, for C.
    3) TCL interface via TCL WINDOW command.
  - Windows could be shared.

CENTURY COMPUTING
DATA SYSTEMS DEVELOPMENT

Chronology (Cont.)

- Bad Points (DEC VAXstation 100, cont.)
  - No icons, scrollbars, buttons or subwindows.
  - No window resizing.
  - "Native Windows" were used resulting in awkward window movement.
  - Had to use undocumented routines.
  - Only VAXstation CORE and TEKTRONIX 4014 graphics were supported.
  - Major "bug" in the VS100 software.
  - DIGITAL discontinued the VAXstation 100.

CENTURY
COMPUTING
DATA SYSTEMS DEVELOPMENT

Chronology (Cont.)

- APOLLO Workstation Prototype
  - First UNIX based machine.
  - First real workstation.
  - Window management software quite different than the VAXstation 100 software.
- Good Points:
  - Nice multi-window development environment.
  - Good documentation and excellent examples.
- Bad points:
  - Window sharing limited to "Parent" and "Child".
  - Base window management system somewhat primative.
  - Interprocess sockets not implemented properly.

CENTURY
COMPUTING
DATA SYSTEMS DEVELOPMENT

Chronology (Cont.)

- "Striping" introduced.
  - Implement easy functions first, then harder.
  - Stripe #1 implemented on the APOLLO system.

CENTURY
COMPUTING
DATA SYSTEMS DEVELOPMENT

Chronology (Cont.)

- SUN 3/160 Workstation.
  - No window passing.
  - SUNVIEW V3.0 documentation is poor and and it contains terrible examples.
  - The SUNVIEW window management software is generally more sophisticated than the other workstations.
    - Popup menus
    - Scrollbars
    - Buttons
    - Icons
    - Subwindows (panes)
  - Decided to implement the TAE Facelift to better understand the capabilities of the SUN.
  - New "window server" model formulated.

CENTURY
COMPUTING
DATA SYSTEMS DEVELOPMENT

A Look into the Future

- VAXstation II/GPX port of the TAE Facelift.
- MACworkstation port of the TAE Facefift. (Gemoets)
- Application/User Interface design tools.
- Continue to look at new developments in the field, such as Carnegie Mellon's Andrew, and MIT's XWINDOWS.
- Object oriented User Interface Management Systems (UIMSs)
- Build on our experience.

CENTURY
COMPUTING
DATA SYSTEMS DEVELOPMENT

Conclusions

- Portability continues to dominate.
- Lack of a industry wide Window Management System Standard.
- Must appreciate the immense variety and complexity of the task.
- Difficulty of window sharing and passing.
- Most portable architecture appears to be one with "Window Servers".

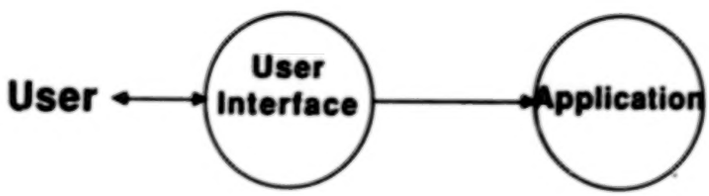
USING TAE WITH DATABASES

Michael Lane Gough, Nathan Lama, James
National Space Science Data Center

PRECEDING PAGE BLANK NOT FILMED

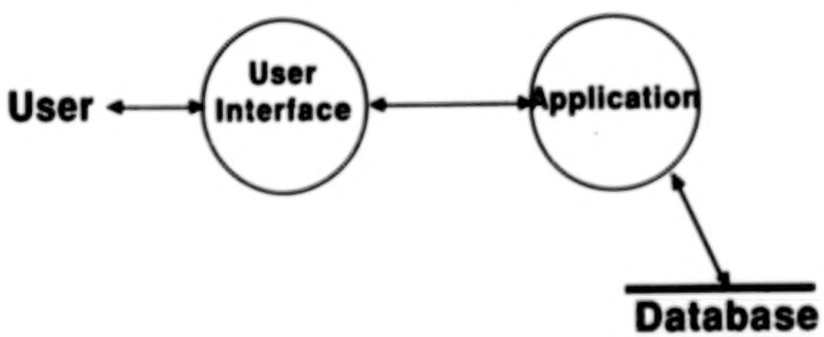
Traditional Applications

- User interfaces are built independently of application software



Database Applications

- Information must flow from the database into the user interface

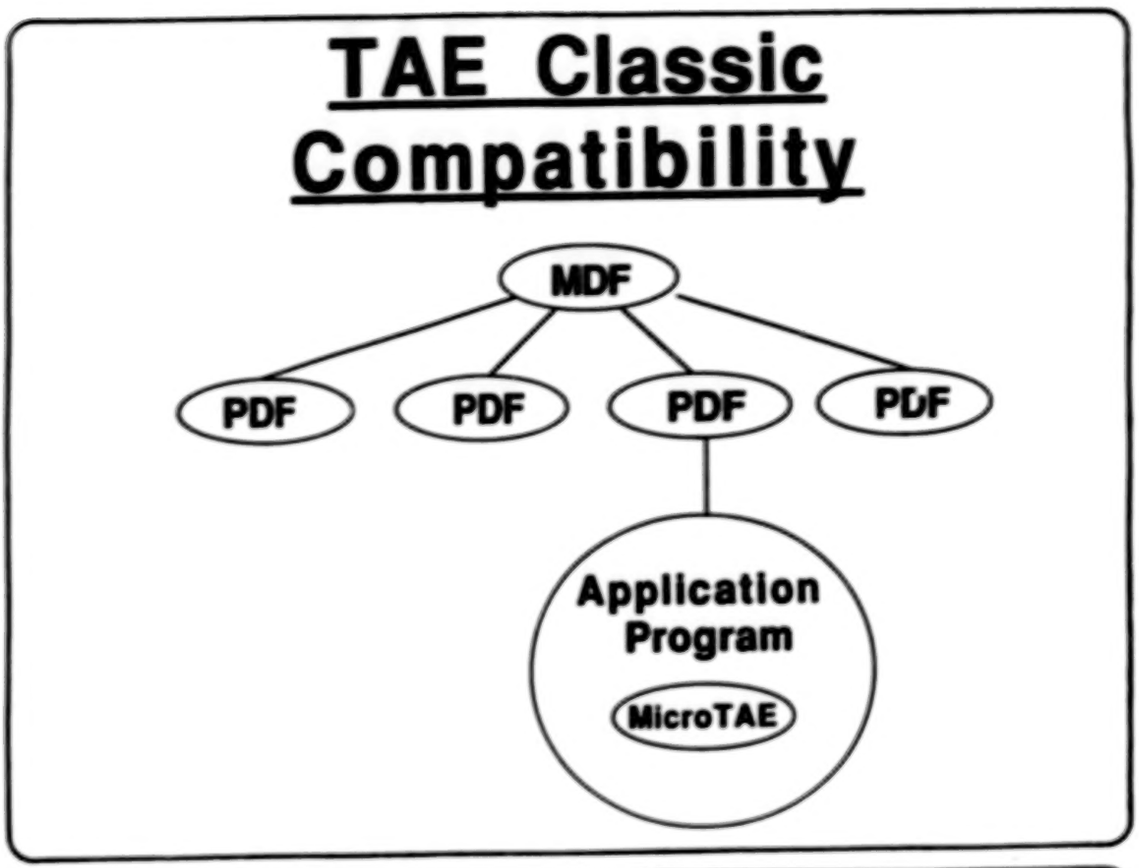


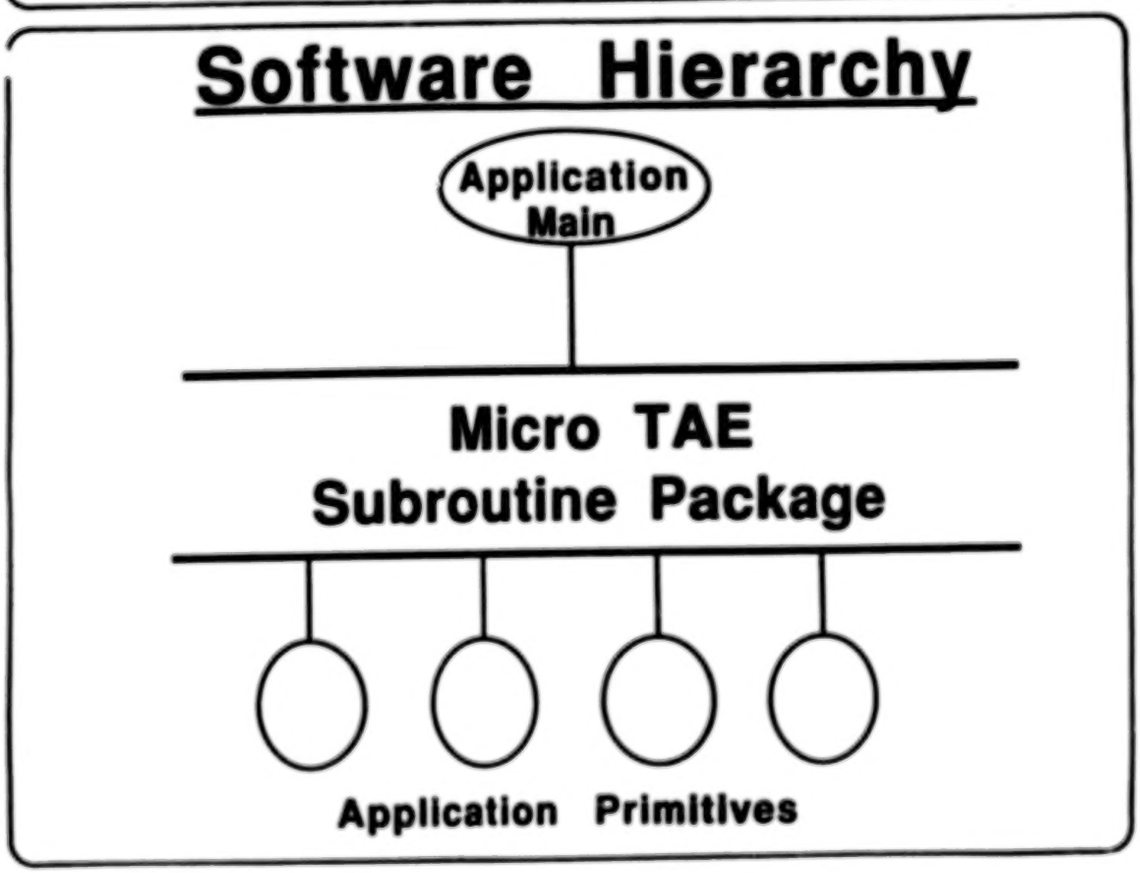
Previous Implementations

- PCDS and CDAW have achieved "Dynamic Tutors" by generating PDFs at run-time
  - very slow
  - clutters user's directory
  - not modular
    - can't run on a PC or IBM mainframe

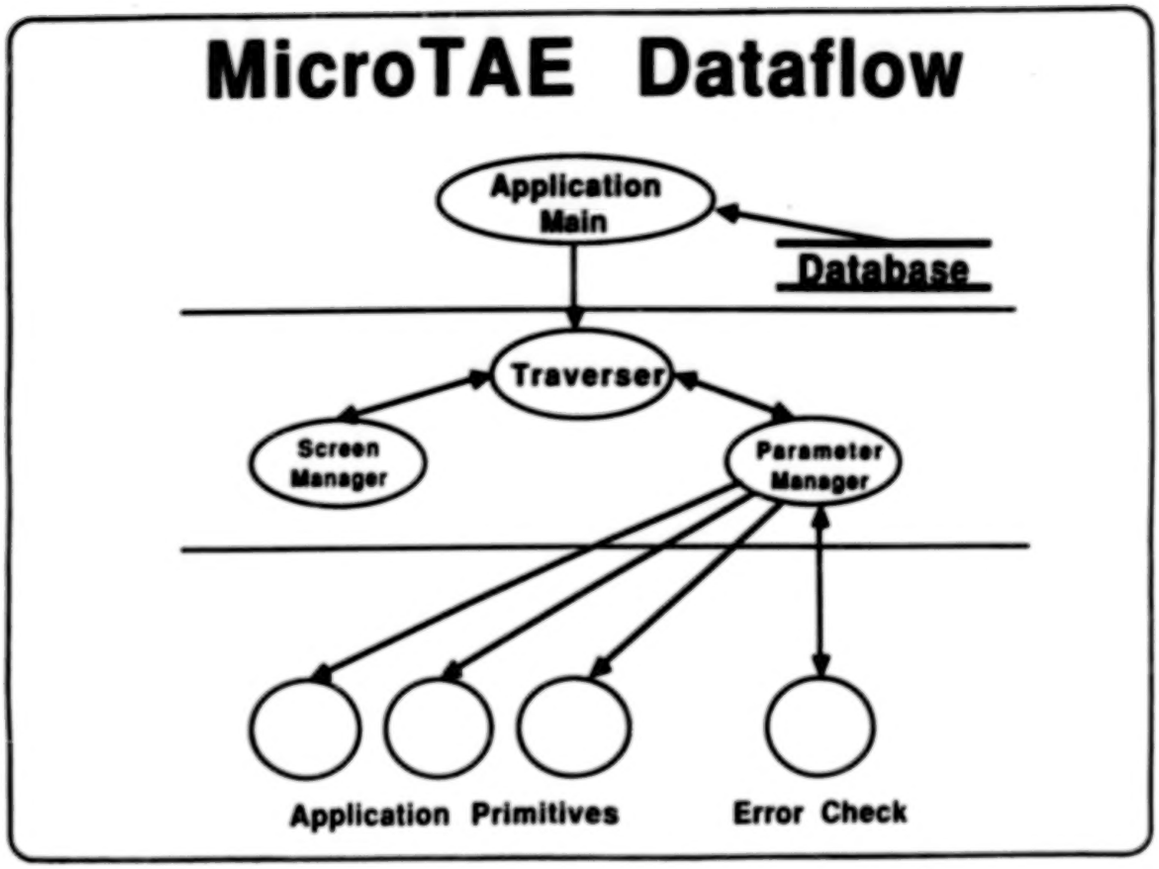
MicroTAE
The Shell Approach

- A fully modular set of subroutines that mimic the operation of TAE Classic
- Can run in tandem with TAE Classic
- Provides a "tighter" connection between the application and the user interface
- Portable to non-multitasking environments
- Fully Reentrant


TAE Classic
Compatibility
MDF
PDF
PDF
PDF
PDF
Application
Program
MicroTAE


Software Hierarchy
Application
Main
Micro TAE
Subroutine Package
Application Primitives

MicroTAE Dataflow



N87
23159
UNCLAS

N87-23159

A DESIGN FOR A NEW CATALOG MANAGER AND ASSOCIATED FILE MANAGEMENT FOR THE LAND ANALYSIS SYSTEM (LAS)

Cheryl Greenhagen
TGS Technology, Inc.
EROS Data Center

PRECEDING PAGE BLANK NOT FILMED

A DESIGN FOR A NEW CATALOG MANAGER AND ASSOCIATED FILE MANAGEMENT FOR THE LAND ANALYSIS SYSTEM (LAS)

Cheryl Greenhagen

TGS Technology, Inc.

EROS Data Center

1 INTRODUCTION

Due to the large number of different types of files used in an image processing system, a mechanism for file management beyond the bounds of typical operating systems is necessary. The TAE Catalog Manager was written to meet this need.

Land Analysis System users at the EROS Data Center (EDC) encountered some problems in using the TAE catalog manager, including catalog corruption, networking difficulties, and lack of a reliable tape storage and retrieval capability. These problems, coupled with the complexity of the TAE catalog manager, led to the decision to design a new file management system for LAS, tailored to the needs of the EDC user community. This design effort addressed catalog management, label services, associated data management, and enhancements to LAS applications.

This paper briefly describes this alternate design for file management of an image processing system.

2 DESIGN GOALS

The following goals were set for the catalog manager design project:

Combine all components to create an integrated system.

Provide a single user interface for both VMS and UNIX.

Increase VMS/UNIX compatibility.

Provide the functional capabilities of the current TAE Catalog Manager, and incorporate the additional capabilities required by local users.

Provide a simple and flexible system, minimizing the amount of code to write and maintain, and minimizing the impact of adding vector, tabular, or other types of data in the future.

Minimize the impact on existing LAS Application Programs.

Minimize system maintenance and support.

Promote data integrity, minimize corruption, and allow several recovery procedures.

Eliminate performance bottlenecks wherever possible.

Simplify transfer of data between computer systems, for both network and tape transfer.

3 CATALOG MANAGER DESIGN APPROACH

Several possible designs for a new catalog manager were considered. The design chosen is a very simple one, based on the host operating system's disk file manager. It provides the same user interface on both VMS and UNIX.

Each user will have his own "catalog", a subdirectory named "CM" in the user's directory. Cataloged disk files are those residing in the user's CM directory.

In addition to the user's files, a set of sequential ASCII files is maintained in the CM directory for catalog manager bookkeeping functions, including tracking cataloged tape files, handling aliases, and corrupt file detection.

The catalog manager design also provides two types of offline storage: short-term and long-term archive tapes.

Files that have been copied to short-term archive tapes are part of the user's catalog and are accessible by the software; these files will be automatically retrieved if requested for input. Information about these files is found in the tape catalog in the user's directory.

Files that have been copied to long-term archive tapes are removed from the user's catalog. No information about these files is stored online.

The catalog manager file name (TAE name) will have the same format on all operating systems. Application programs accept and display the TAE name, so the user can move from one system to another without changing the TAE naming convention, and without being familiar with the host operating system file name. The TAE user does not see the host name except by explicit request.

The new catalog manager will translate the TAE name to a recognizable host name - in fact, there is a close correspondence between the TAE name and the host name. This makes it easier to use operating system tools and other non-TAE software to process cataloged files.

The catalog manager file name has the following format:

```
[user.directory1.directory2 ...]filename;extension
```

The username and directory name specification is enclosed in square brackets. "User" is any valid username on the system. The directory names may contain alphabetic characters and numeric characters. The username and directory names are separated by periods. The filename and extension name follow. They may contain alphabetic characters, numeric characters, and periods (".") . The filename and extension name are separated by a semicolon.

Each of the components (directories, filename and extension) of the catalog manager name may be up to 39 characters long. The whole name may be up to 248 characters long.

The username and directory name specification is optional. If not specified, the current directory is assumed. By using the full file specification, the user may access any file in any directory, subject to the file protections of the operating system.

The following example illustrates TAE name - host name correspondence:

```
TAE name:       [SMITH.NY.MSS]STRETCHED.IMAGE;HIS

VMS host name:  [SMITH.CM.NY.MSS]STRETCHED_IMAGE.HIS

UNIX host name: /smith/cm/ny/mss/stretched_image.his
```

4 CATALOG MANAGER UTILITIES

The current TAE catalog manager is a constantly-running program. By contrast, the new catalog manager is just a collection of utilities and support routines. A brief overview of these utilities follows.

4.1 Create, Delete, And List Aliases

The alias utilities allow the user to create (assign), delete, and list aliases.

An alias is just a short name for another string - the alias text. The use of aliases is supported in Catalog Manager program parameters. When an alias name is used, as a parameter or part of a parameter, the alias text is substituted. Each user will have his own set of defined aliases.

Alias information will be kept in an ASCII text file in the user's CM directory. The alias name is a 9-character string beginning with "$". The alias text is a 255-character string. In order to make the file editable by EDT, which does not handle records longer than 255 characters, the file will consist of pairs of records. The first record in every pair will contain the alias name; the second record of the pair will contain the alias text.

The following example illustrates the alias file.

```
$WINDOW
"(200,200,100,100:2,3,4)"

$IN
"NEWYORK.IMAGE1(:1,2) + NEWYORK.IMAGE2(:3)"
```

4.2 Copy, Rename, Delete, And List Files

The catalog manager includes utilities to copy, rename, delete, and list cataloged files. The list utility includes the capability to list selected file attributes (e.g., file creation date), or image attributes (e.g., number of bands, data type) in addition to the file name.

4.3 Create, Delete, Display, And Set Directories

The directory utilities allow the user to create and delete directories in his directory tree, and to display or set his current, or "working", directory.

4.4 Copy Files To And From Archive Tapes

The tape-handling utilities allow the user to copy cataloged files to short-term or long-term archive tapes, and to retrieve files from archive tapes.

The physical format of the archive tapes will follow an ANSI standard, providing convenient multiple tape handling and usability on many systems.

The catalog manager will generate some archive information in addition to the ANSI label information.

The catalog manager information for archive tapes includes the file name (TAE name), file status, a tape library identifier, the creation timestamp of the file, the size of the file, the tape length and density, and the last access date of the tape.

For short-term archive tapes, the catalog manager stores this information about the tape and the files in a sequential ASCII file in the user's CM directory.

For long-term archive tapes, the same information is written to the end of the archive tape. No information is stored on-line for long-term archive tapes.

4.5 Support Functions

The catalog manager support utilities include statusing utilities and tape management utilities for use by the computer operations staff.

5 SYSTEM-LEVEL ENHANCEMENTS

5.1 Associated Files

In the course of processing raster images, a user may create several types of associated data that should reside with the image for easy reference. This information will be stored in a set of files associated with the image. Label information for the image will also be stored in associated files.

Files will be associated implicitly, by filename. All files in a directory with the same filename, but different extensions, are considered to be associated.

The existence of implicitly-associated separate files for the label information and associated data gives great flexibility, simplifies the task of accessing label information, and allows the image and its associated files to be easily treated as a "data set" for tape archive and retrieval, and file transfer.

Associated files will have records of variable length and different data types. Wherever possible, however, these files will be sequential in organization and will share a common feature: each record will begin with 3 ASCII fields of known size, containing the length of the record, the data type of the record contents, and a key.

Any extension may be used for an associated file. However, some standard extensions will be used in naming frequently used types of files:

DDR	Data Descriptor Records File
CWT	Convolution Weights File
DPF	Display Parameter File
POINT	Graphics Overlay Point File
LINE	Graphics Overlay Line File
POLY	Graphics Overlay Polygon File
ANNOT	Graphics Overlay Annotation File
HIS	History File
IMG	Image File
LUT	Look-up Table File
STAT	Statistics File

5.2 Input File Handling

If an application program attempts to access a file that is not online, the application program will automatically attempt to retrieve the file from the user's short-term archive tapes. If successful, the

file will remain on disk when processing completes. However, a user may not access files from another user's archive tapes.

5.3 Output File Handling

The EDC user community requires that cataloged files be unique in the catalog - the same file may not exist both on disk and on tape.

To ensure uniqueness, before opening a new output file, application programs will check both the disk catalog and the store tape catalog to ensure that the user's catalog does not already contain a file by the specified name. If a specified output file name already exists in the catalog, an error message will be generated, and the new file will not be created.

5.4 Corrupt File Detection

Corrupt file detection will be implemented by means of a convention followed the application programs. Whenever a file is opened for write access, an identifying entry is written into a corrupt file list in the user's CM directory. The entry will be deleted when the file is closed. If the program aborts or the system crashes before the file is closed, the entry will remain in the corrupt file list. A file found in the corrupt file list when no application program is using it is assumed to be incomplete, and may be replaced.

5.5 Aliases

LAS applications will be enhanced to handle aliases in parameters, by simple string substitution.

5.6 Image I/O Libraries

Currently, three different methods of reading and writing image files exist in different software systems at EDC:

TAE I/O, based on the XI package from Century, is available on both VMS and UNIX. TAE I/O stores all bands for an image in one file.

LAS I/O, based on VMS RMS I/O, is available on VMS only. LAS I/O stores each band in a separate file.

NEWLAS I/O, written at EDC and based on TAE I/O, is available on both VMS and UNIX. NEWLAS I/O stores all bands for an image in one file.

NEWLAS I/O is EDC's target image processing system. Until the target is reached, application programs will provide a consistent "virtual image" approach across all of the image processing systems. This means that the user will always view an images as a cohesive set. The applications will automatically access the correct host file(s) for the user-specified catalog name, isolating the user from the physical location and configuration of the image bands.

5.7 Build PDF From History File

A program will be provided to create a Proc Definition File from a history file.

6 SUMMARY - SYSTEM ENHANCEMENT BENEFITS

The new file management design will provide the many benefits, including improved system integration, increased flexibility, enhanced reliability, enhanced portability, improved performance, and improved maintainability.

N87
23160
UNCLAS

N87-23160

NASA REGIONAL PLANETARY IMAGE FACILITY
IMAGE RETRIEVAL AND PROCESSING SYSTEM

Susan Slavney
Washington University

NASA REGIONAL PLANETARY IMAGE FACILITY
IMAGE RETRIEVAL AND PROCESSING SYSTEM

- RPIFs were set up by NASA to house and maintain image data produced by space probes to moons and planets.
- There are currently nine RPIFs in the U.S. and four in Europe.
- In the near future, RPIFs will provide access to digital planetary image data.
- Washington U. and USGS Flagstaff are jointly developing a MicroVAXII based system to manage and analyze planetary image data.

NASA RPIF IMAGE WORKSTATION

- Workstation main functions will include:
  - Search through database
  - Retrieval of digital images
  - Processing and display of digital images
- The MicroVAX effort is being done under the auspices of the Planetary Data System (PDS).
- The workstation will be used as a prototype RPIF image processing system in PDS Build One.
- The workstation will eventually be replicated at other RPIFs.

DESIGN CONSIDERATIONS

Types of information to be stored:

- General information about planets and planetary exploration
- Information about specific data sets

Types of data to be stored:

- Digital images on magnetic tape or CD
- Non-digital data such as photographic prints and maps

Types of users:

- Experienced users such as RPIF staff and scientists
- Occasional, knowledgeable users such as visiting scientists
- General public such as high school science teachers.

DESIGN CONSIDERATIONS

Data presentation:

- View single entries or tables of data
- Output to terminal, file or printer
- Some simple graphics

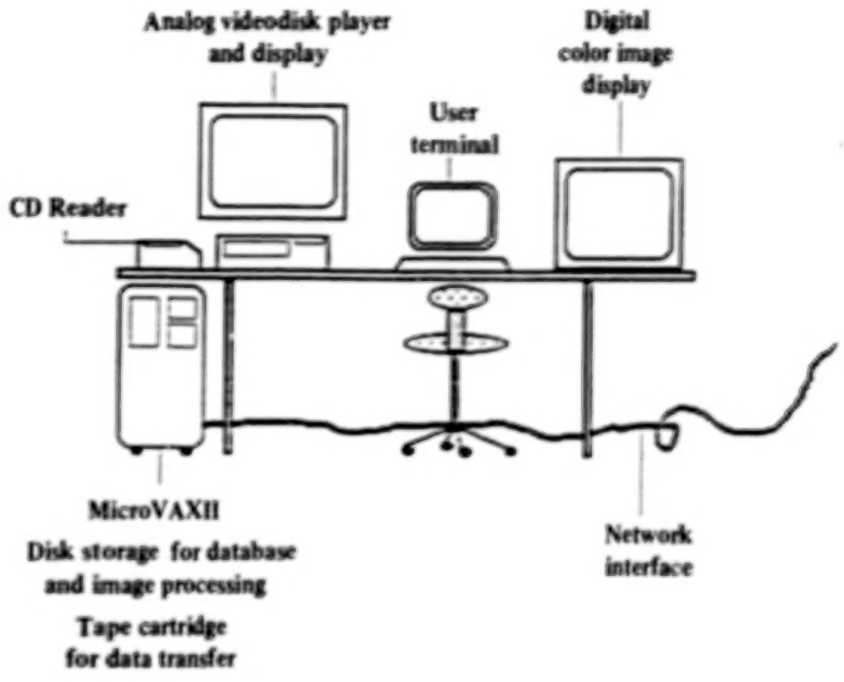
Browse capability

Access to digital data

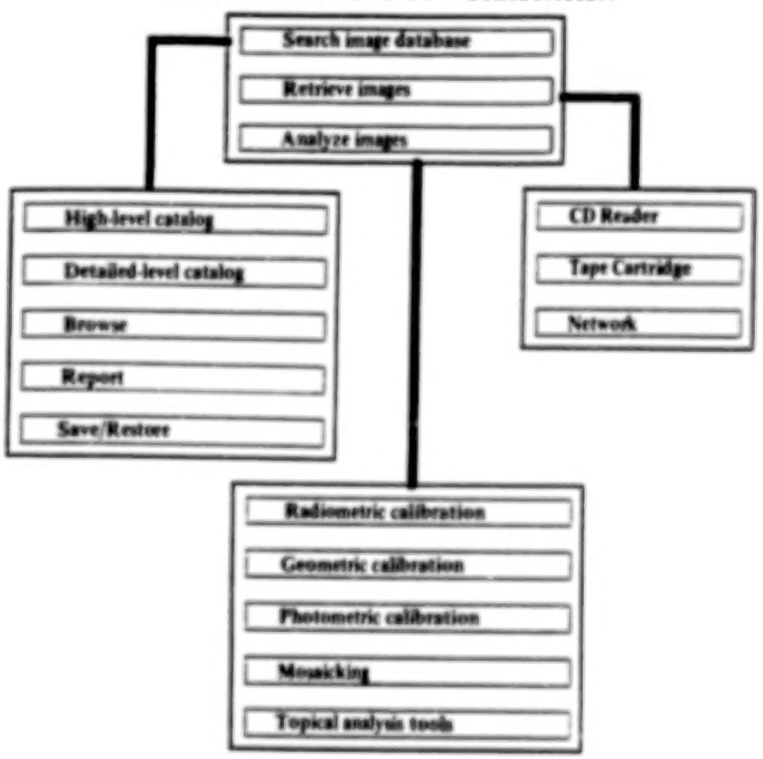
Image processing capabilities:

- Radiometric and geometric calibration
- Color image display

WORKSTATION COMPONENTS



MAJOR FUNCTIONS OF WORKSTATION



DATABASE DESIGN

The database is divided into **high-level** and **detailed-level** catalogs.

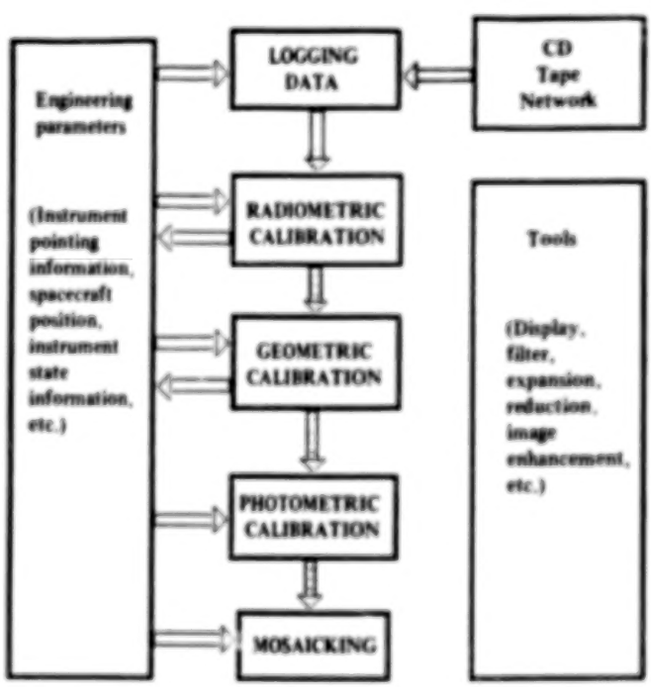
The high-level catalog is used to find out about planets and planetary exploration. It contains information on:

- planetary missions
- spacecraft
- instruments
- planets

The detailed-level catlog is used to find data to be analyzed. It contains information on:

- engineering parameters for images (picture location, filter, etc.)
- maps produced from planetary images
- mosaics of planetary images
- location and format of digital data

IMAGE ANALYSIS FUNCTIONS



Software developed by USGS Flagstaff

USES OF TAE IN WORKSTATION

- Menus used for main functions
- File access and image processing programs use TAE tutor screens to receive parameters from user, and TAE subroutines to pass parameters to programs.
- Database access and reporting programs use TAE tutor screens in conjunction with commercial database forms package
- PDS is evaluating TAE as a possible user interface for Build One

CURRENT STATUS OF DEVELOPMENT

- System hardware installed, including hard disk for image storage, CD-Reader for image retrieval, digital image display, and analog videodisk player and monitor.
- TAE and database management software installed on MicroVAXII. Installation of image processing software in progress.
- Database design for high-level and detailed-level catalogs in process
- Developed test programs to read and display digital images from CD.
- Prototype workstation expected to be completed by March 1987.

TAE V1.4 OVERVIEW

Phil Miller
Century Computing, Inc.

October 2, 1986

General

- General philosophy: bug fixes and minor enhancements
- New features:
  - Menu Captive and TCL Captive
  - RUNTYPE=ASYNC-PROCESS
  - Inter-proc Communication
- For now: VMS only (4.2 or greater)
- TAE/UNIX V1.4 scheduled for early CY 1987
- No documentation release
- Complete documentation re-release: early CY 1987
- Performance: neutral
- In general, re-LINK of all applications required

Menu Captive

- A security feature
- Prevents menu user from using TCL
- Remember to disable VMS CONTROL/Y in LOGIN.COM

```
!
! Sample SLOGON.PDF
!
PROCEDURE OPTIONS=NOINTERRUPT     !cannot abort SLOGON
BODY
DISABLE-INTERRUPT          !no TCL allowed; auto abort
MENU-CLOSED  MYROOT.MDF    !cannot leave menu
END-PROC
```

TCL Captive

- New TCL command: DISABLE-HOST
- Makes DCL and EXIT commands invalid
- There is no ENABLE-HOST
- Remember to disable VMS CONTROL/Y also

```
!
! Sample SLOGON.PDF
!
PROCEDURE  OPTIONS=NOINTERRUPT
BODY
DISABLE-HOST
END-PROC
```


ASYNC-PROCESS

- ASYNC job submitted from TCL:

```
procname |RUNTYPE=ASYNC-PROCESS| parameters...
```

- Much like normal ASYNC job but no associated TM
- Provides for efficient ASYNC execution.
- The proc must be a PROCESS-type proc.
- Several other restrictions exist, e.g, no dynamic tutor

SENDVAR and RECVAR

- SENDVAR: send VBLOCK to another job
- RECVAR: receive VBLOCK from another job
- The "another job" may be ASYNC or ASYNC-PROCESS
- A basic capability:
  - RECVAR blocks
  - One input queue per job
  - No select or events
  - No ASTs
  - No CONTROL/C interruption


```
!
! Simple "server" in TCL.
!
! Run this with |RUNTYPE=ASYNC| qualifier.
!
PROCEDURE
    REFGBL $PARENT              !PARENT'S JOB NAME
    LOCAL _JOB STRING           !NAME OF SENDER
    LOCAL VALUE INTEGER
BODY
ENABLE-RECVAR                   !ENABLE SERVER QUEUE
SENDVAR JOB=@$PARENT            !TELL PARENT WE'RE READY
LOOP
    RECVAR (_JOB, VALUE)        !RECEIVE REQUEST
    LET VALUE = VALUE + 1       !OPERATE ON REQUEST
    SENDVAR VALUE JOB=@_JOB     !RETURN TO REQUESTOR
END-LOOP
END-PROC
```

NEW PORTS OF TAE

PC TAE

PORTING TAE TO THE IBM PC

Steven Engle and Brian Phillips
NASA/Ames Research Center

PRECEDING PAGE BLANK NOT FILMED

RATIONALE

To provide scientists who are using the

Pilot Land Data System

with a TAE interface for their "PC"

The TAE environment is supported on the larger PLDS computers

for accessing and processing large digital image data sets.

CHOICES

A Run TAE under local host operating system

MS-DOS, PC-DOS, ...

B Run TAE under PC hosted operating system

PCUNIX, PCVMS, SYSTEM V/AT, ...

C Run TAE on a PC attached coprocessor

SRITEK 6800 SYSTEMV/68

PORTING CHOICES

What Hardware to select ?

Which C compiler to use ?

On which TAE version to base port ?

AVAILABLE RESOURCES

Shared PC/XT with 750 KB and 30 MB

Microsoft C compiler _ version 2.00

VAX/VMS with "VMS" TAE source

Kermit connection to VAX

PC tools _ Edlin, Link, etc

INITIAL CHOICE

Use Microsoft C compiler with selected "VMS" TAE modules.

["asynch" and "remote" modules discarded]

PRESENT STATUS

Selected Modules compile and link.
Object size > 310KB.

Incomplete _ Nonportable modules added.
Object runs to "logon".pdf.

Non-portable modules

string.cnp:	C routines used, no assembly.
file.cnp:	Unix version used, needs changes.
terminal.cnp:	Line at a time, will change.
termattn.cnp:	Needs integration with events.
process.cnp:	Not yet started.

OBSERVATIONS

With machine dependent portions of the TAE system isolated into non-portable modules, their implementation can require deep knowledge of the odd behaviour of the target machine.

Terminal I/O requires both knowledge of the compiler, its I/O support library, and the side effects of the BIOS.

Both with pointers requiring 32 bits, and with integers, longs, and doubles all being returned differently, then everything must be declared correctly.

Structures require correct padding, and NULL used correctly

Support tools vastly improve the job both in time for testing, correction and analysis, and in less frustration.

Time for testing is essential.

COMMENTS

COMPILER

The more sophisticated the code,
the more important the C compiler

[Microsoft C replaced by Computer Inovations C86]
Aztec C just received

TOOLS

Adequate Tools are essential

Edlin is a dog, now use 2, egrep, and epsilon

Make, Xrefs , Lint, needed

HARDWARE

XT disk overloaded and compiles are slowww....

Now use AT with 140MB disk but only 250 KB

COMMUNICATIONS

Kermit good, FTP added

Compiler

Compilers are good at picking up the simple errors and the non-portable coding practices.

Lots of "VMS" debris was picked out.

```
#include    tae$olb:terminal.c*p
globalref   long  dm_bytes;
```

Size changes

```
if (tmval2.uval.intval && tmp1.uval.realval)

   intval  type  TAEINT  = int

   realval  type  TAEFLOAT  = double
```

Compiler peculiarities

String initializers for character arrays in structures should be enclosed in curly braces {}.

Forward references cannot be handled because of unknown size

```
static    COUNT   geofs ();
.
.
.  ( code )
.
.
static   FUNCTION   COUNT   geofs(parm1, parm2);
```

Heartburn (illustration only)

```
TEXT        * nullstr[]   =   "";
TEXT        * all[]       =   "ALL";

struct      RESIDVAR   {
            char        name[NAMESIZ - 1];
            TINY        tiny;
            unsigned    kiv : 1;
            TINY        minc;
            TINY        maxc;
            COUNT       size;
            TINY        dcount;
            GENPTR      valid;
            GENPTR      dvp;    };

struct      TUTCMD     {
            TINY        abchar;
            TEXT        cmdname[NAMESIZ - 1];
            TINY        numpar;
            unsigned    parm_allowed  : 1;
            unsigned    subc_allowed  : 1;
            CODE          (*tutfunc)();    };


static      struct     RESIDVAR   name[]   =
      {
"COMMAND",   V_STRING,0,1,1,   SIZ,-1,NULL,(GENPTR)nullstr,
"VARIABLE",  V_STRING,0,1,MAXVAL,NAMESIZ,1,NULL,&all
      };

static      struct     TUTCMD   tomd[]   =
      {
0,   "-",       0, FALSE,  TRUE,  tutsubeq,
0,   "parm=",   0,  TRUE,  FALSE,  tutprmeq
      };
```

APPLE MACINTOSH COMPUTER AS A TAE WINDOW MANAGER

Richard Gemoets
Appaloosa System

APPLE MACINTOSH COMPUTER AS A TAE-PLUS WORKSTATION

- INTRODUCTION
- THE WORKSTATION CONCEPT
  - WHY THE WORKSTATION APPROACH?
  - SALIENT CHARACTERISTICS
  - EXAMPLES OF USE
  - PITFALLS
- TAE WORKSTATION EXPERIENCE
- MacTAE DEVELOPMENT
- INSIDE MacTAE
  - SYSTEM ARCHITECTURE DIAGRAM
  - STRUCTURE CHART W/ TAE CONNECTIONS
  - MACWORKSTATION FUNCTIONALITY
  - DEVELOPMENT ENVIRONMENT
- OUTSIDE MacTAE
  - PRIMARY MENUS AND SCREENS
  - DIFFERENCES FROM SUN-TAE
  - PORTING CONSIDERATIONS
  - OTHER SALIENT CHARACTERISTICS
- "THE GOOD, THE BAD, & THE UGLY"
  - ON THE POSITIVE SIDE
  - A FEW NEGATIVES
  - BUGS AND QUIRKS

APPLE MACINTOSH COMPUTER AS A TAE-PLUS WORKSTATION

- THE WORKSTATION CONCEPT
  - WHY THE WORKSTATION APPROACH?
    - ✓ COST EFFECTIVE
    - ✓ MULTI - PURPOSE
    - ✓ INCREASED PRODUCTIVITY
    - ✓ REDUCE USER RE-TRAINING
    - ✓ DISTRIBUTE THE LOAD
  - SALIENT CHARACTERISTICS
    - ✓ LOW COST ($2-3K RANGE BEST)
    - ✓ HUMAN ENGINEERED USER XFACE
    - ✓ BIT-MAPPED GRAPHICS
    - ✓ HIGH PERFORMANCE
    - ✓ RUNS VARIETY OF APPLICATIONS
    - ✓ LOW POWER CONSUMPTION
    - ✓ DESKTOP SIZE
  - EXAMPLES OF USE
    - ✓ ENGINEERING ANALYSIS
    - ✓ ENGINEERING DESIGN
    - ✓ NAS FACILITY AT ARC
    - ✓ COMMAND & CONTROL

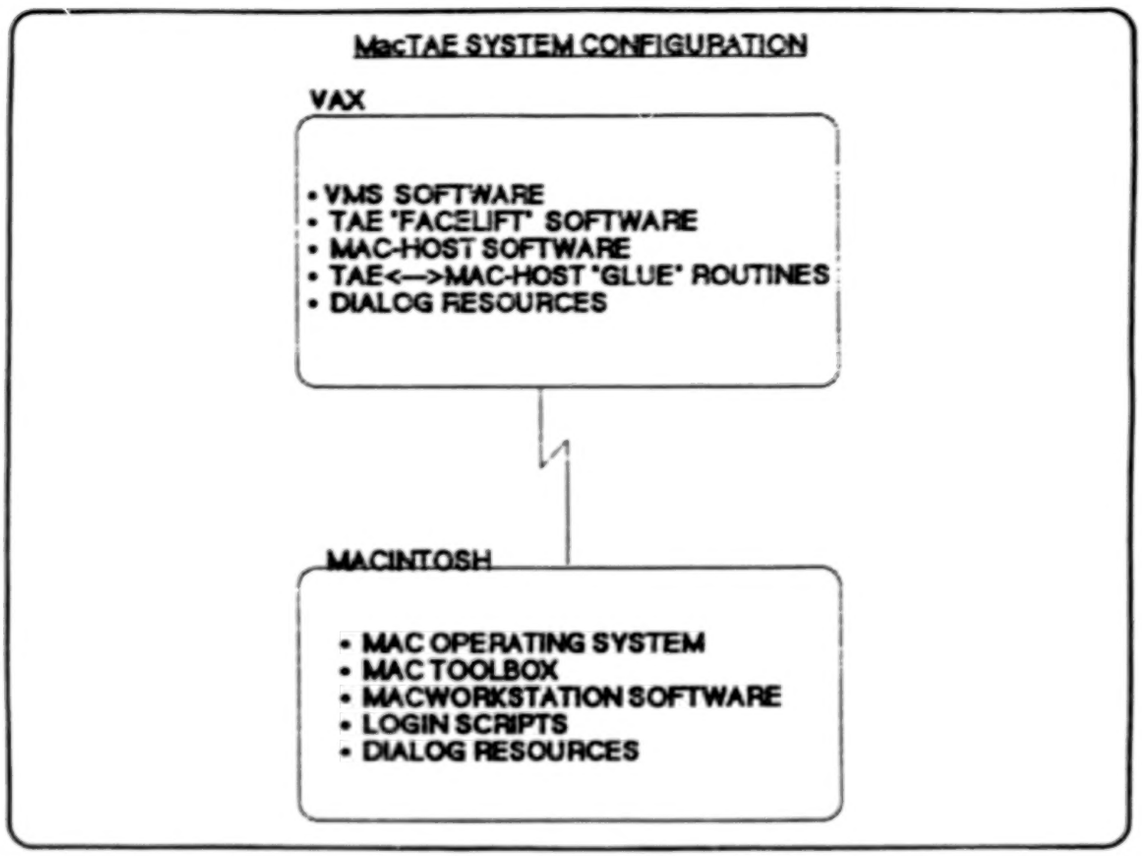
APPLE MACINTOSH COMPUTER AS A TAE-PLUS WORKSTATION

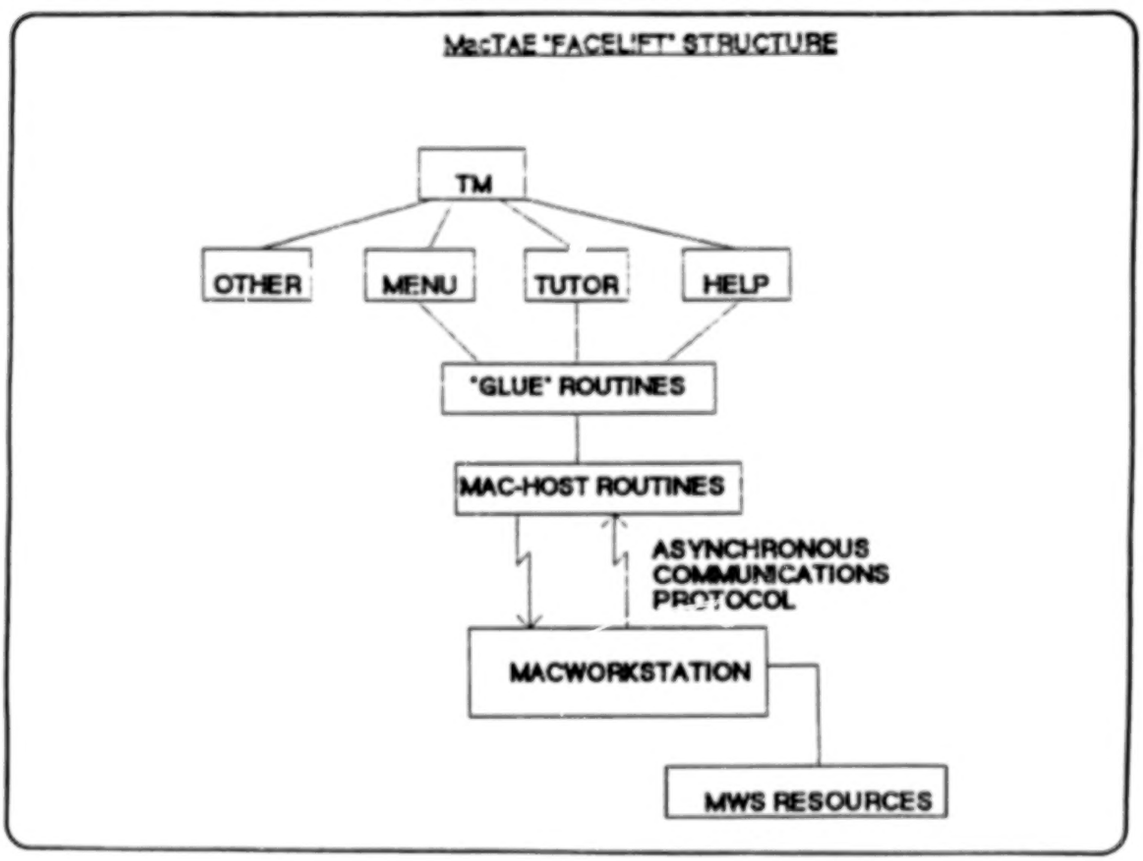
- MacTAE DEVELOPMENT
  - MacWorkStation EVALUATION
    - ✓ JUST *GETTING* THE DARN THING WAS HALF THE BATTLE!
    - ✓ WORK COMPLETE: "*LOOKSA OK*"
  - DEVELOP MINI-APPLICATION
    - ✓ "PROOF OF CONCEPT
    - ✓ WORK COMPLETE: SEE DEMO
  - DEVELOP VS100 VERSION
    - ✓ EFFORT SCRAPPED 8/86
    - ✓ DEAD-END PATH
  - DEVELOP TAE "FACELIFT" VERSION
    - ✓ TAE1.3b + SUN ENHANCEMENTS
    - ✓ WORK IN PROGRESS
  - DEVELOP TAE - PLUS VERSION
    - ✓ WORK IN PLANNING
  - DEVELOP UIMS PROGRAMMER TOOLKIT
    - ✓ NOT FUNDED
  - FULL PORT TO MACINTOSH (UNIX)?!?

APPLE MACINTOSH COMPUTER AS A TAE-PLUS WORKSTATION

- PITFALLS
  - ✓ MORE HARDWARE TO BREAK DOWN
  - ✓ MORE INCOMPATIBLE SOFTWARE
  - ✓ MORE INDECISION (TOO MANY CHOICES)
  - ✓ LACK OF STANDARD USER XFACE
  - ✓ PORTABILITY PROBLEMS

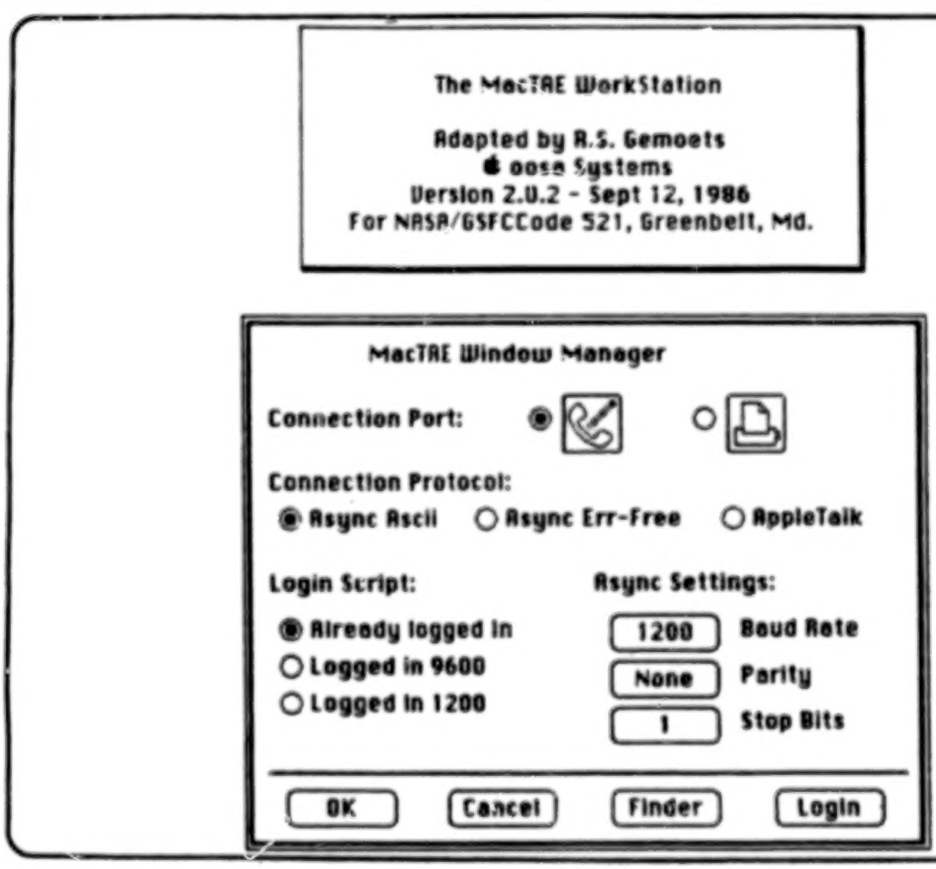
- TAE WORKSTATION EXPERIENCE
  - ✓ DEC VT220
  - ✓ DEC VAXSTATION 100
  - ✓ APPLE MACINTOSH

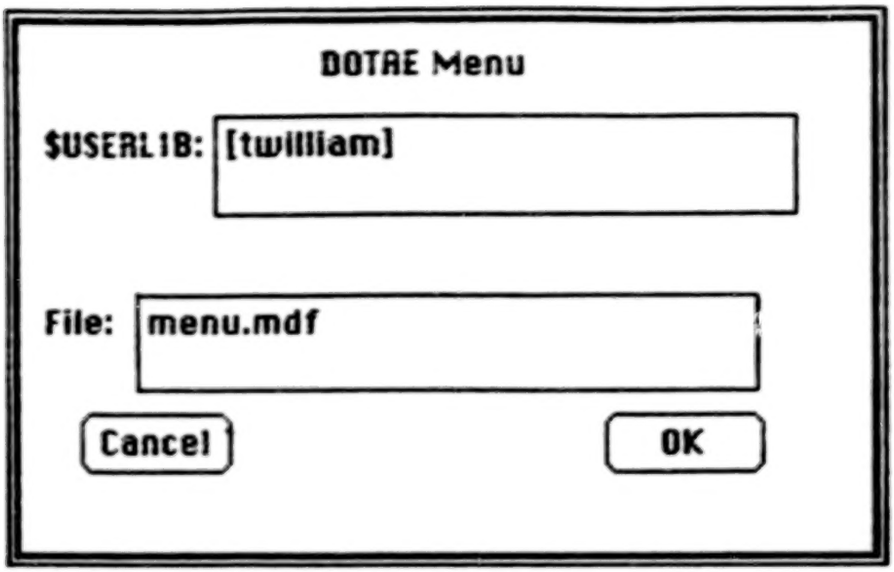

MacTAE SYSTEM CONFIGURATION
VAX
• VMS SOFTWARE
• TAE "FACELIFT" SOFTWARE
• MAC-HOST SOFTWARE
• TAE<—>MAC-HOST "GLUE" ROUTINES
• DIALOG RESOURCES
MACINTOSH
• MAC OPERATING SYSTEM
• MAC TOOLBOX
• MACWORKSTATION SOFTWARE
• LOGIN SCRIPTS
• DIALOG RESOURCES

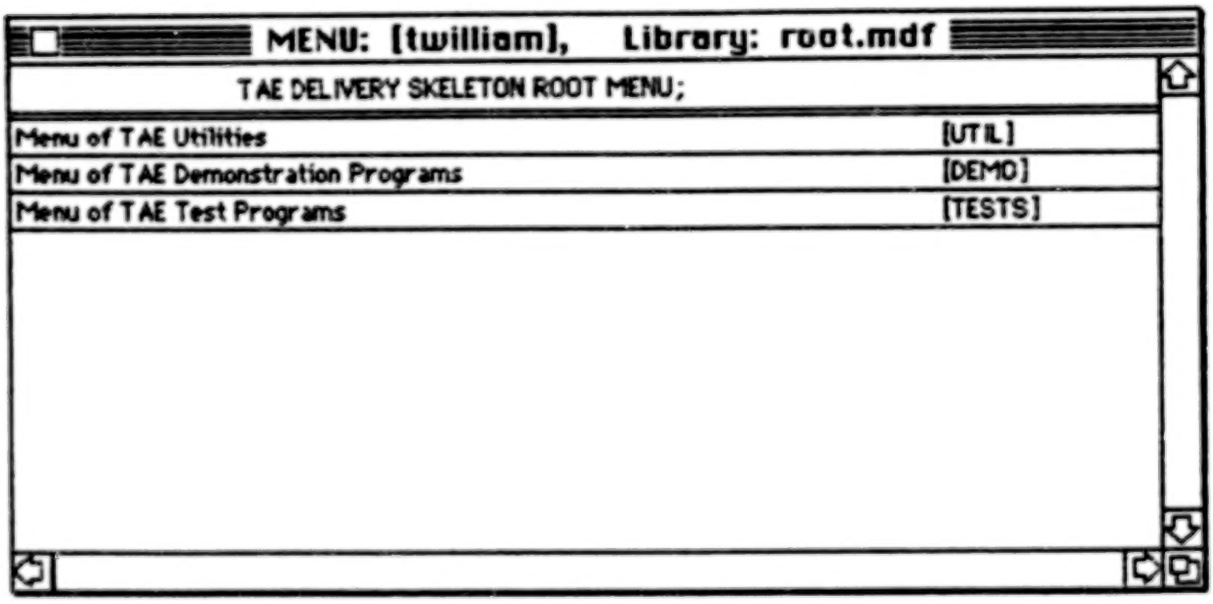

MacTAE "FACELIFT" STRUCTURE
TM
OTHER
MENU
TUTOR
HELP
"GLUE" ROUTINES
MAC-HOST ROUTINES
ASYNCHRONOUS
COMMUNICATIONS
PROTOCOL
MACWORKSTATION
MWS RESOURCES

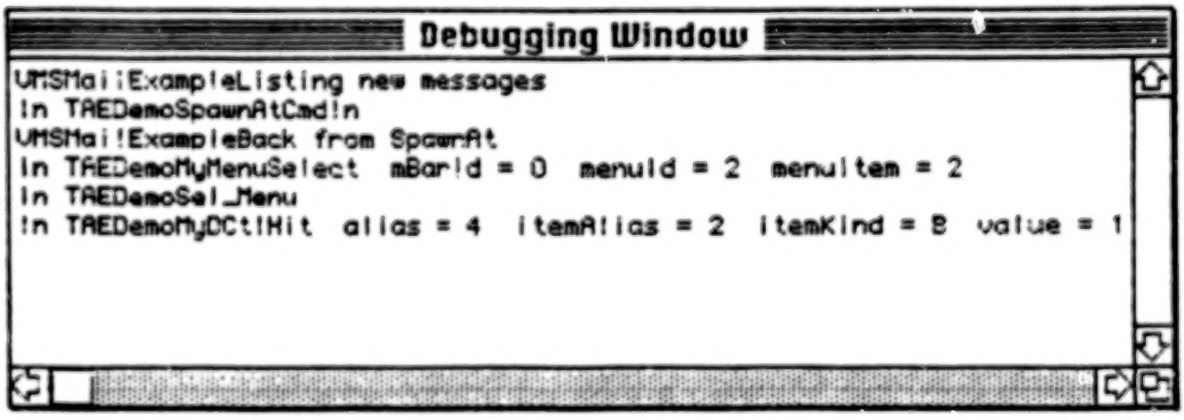
APPLE MACINTOSH COMPUTER AS A TAE-PLUS WORKSTATION

- INSIDE MacTAE
  - + DEVELOPMENT ENVIRONMENT
    - ✓ MAC HOST INTERFACE IN APPLE "C"
    - ✓ TAE IN CENTURY "C"
    - ✓ MACWORKSTATION IN LISA PASCAL
    - ✓ USE DEBUGGING WINDOW AND LOG FOR TRACE INFORMATION
    - ✓ DEBUGGER USE REQUIRES:
      - ⇨ SYSPRV PRIVELEGE
      - ⇨ TERMINAL + MACINTOSH
      - ⇨ 2 TERMINAL LINES
      - ⇨ SOME UNUSUAL LOGICAL ASSIGNMENTS
    - ✓ HANGS REQUIRE KILLING THE JOB
    - ✓ HOST ABORTS DON'T KILL MAC JOB
  - + FUNCTIONS THAT DON'T INVOLVE THE VAX
    - ✓ LOCAL PRINTING
    - ✓ TEXT EDITING
    - ✓ MENU MANAGEMENT
    - ✓ WINDOW MOVEMENT & SIZING
    - ✓ SCROLLING
  - + ADDITIONAL SOFTWARE REQUIRED:
    - ✓ KEYBOARD EVENTS & RESOURCES




DOTAE Menu
$USERLIB: [twilliam]
File: menu.mdf
Cancel
OK


MENU: [twilliam], Library: root.mdf
TAE DELIVERY SKELETON ROOT MENU;
Menu of TAE Utilities [UTIL]
Menu of TAE Demonstration Programs [DEMO]
Menu of TAE Test Programs [TESTS]


Debugging Window
VMSMailExampleListing new messages
In TAEDemoSpawnAtCmdIn
VMSMailExampleBack from SpawnAt
In TAEDemoMyMenuSelect mBarId = 0 menuId = 2 menuItem = 2
In TAEDemoSel_Menu
In TAEDemoMyDCtlHit alias = 4 itemAlias = 2 itemKind = 8 value = 1

APPLE MACINTOSH COMPUTER AS A TAE-PLUS WORKSTATION

- OUTSIDE MacTAE
  - DIFFERENCES FROM SUN -TAE
    - ✓ MENU AND TUTOR FUNCTION BUTTONS ON MENU BAR
    - ✓ SCROLLER ON RIGHT
    - ✓ ERROR LINE IS ALERT DIALOG
  - PORTING CONSIDERATIONS
    - ✓ PROBLEMS WITH FUZZY BASELINES
    - ✓ FAIRLY STRAIGHT-FORWARD (SO FAR)
    - ✓ NO 2 USER INTERFACES WILL HAVE *EXACT* MATCH
    - ✓ PROBLEMS INTEGRATING MULTIPLE , ATONOMOUS SUBSYSTEMS
    - ✓ NO ICON MANAGEMENT
    - ✓ MUST ADD KEYBOARD EVENTS
    - ✓ NO STANDARD GRAPHICS XFACE
  - OTHER SALIENT CHARACTERISTICS
    - ✓ SEPARATE "LOGIN" PROCEDURE
    - ✓ NO DIRECT APPLICATION INTERFACE TO MACWORKSTATION FUNCTIONS
    - ✓ MAC W/S CAN BE DIAL-UP

APPLE MACINTOSH COMPUTER AS A TAE-PLUS WORKSTATION

- "THE GOOD, THE BAD, & THE UGLY"
  - ON THE POSITIVE SIDE
  - A FEW NEGATIVES
  - BUGS AND QUIRKS

MACWORKSTATION EVALUATION

"ON THE NEGATIVE SIDE"

- APPLE SUPPORT IS VIRTUALLY NON-EXISTANT
  (e.g., media for sources, telephone support)
- ALMOST IMPERATIVE TO KNOW OR BE AN INSIDER
- NOT CLASS 1 SOFTWARE OR EVEN A PRODUCT
  (as advertised)
- COMMUNICATIONS SOFTWARE (@$%&*&?*&)

  -LOGIN SCRIPT MAKER – NONE AT FIRST
  -LOGIN PLAYER – CUMBERSOME, PRIMITIVE
  -TIMING SENSITIVE (e.g., no SWITCHER)
  -I WASTED TOO MUCH TIME HERE
- MANY "BUGS & QUIRKS"
  (more on this later)
- MUCH DOCUMENTATION NOT AVAILABLE
  (e.g., no "Inside MacWorkstation" and very little on the MAC side)
- DEBUGGING DIFFICULT
  -no debugger, but fair trace mechanism
- NO ICON MANAGEMENT
- WILL BE DIFFICULT TO "ROBUSTIFY"
- GOOD WORKING KNOWLEDGE OF I.M. ESSENTIAL
- DIALOG MANAGER HARDEST TO USE
  (but it is the MOST used!)
- MAY HAVE TO IMPLEMENT A MINI-DBMS FOR KEEPING TRACK OF MENUS & DIALOGS

MACWORKSTATION EVALUATION

"ON THE POSITIVE SIDE"

- RICH FUNCTIONALITY
- PERFORMS MOST MAC INTERFACE FUNCTIONS
  (missing, e.g., ICON management)
- APPEARS RESPONSIVE
  (perceived delays were from VMS spawns)
- IT *DOES* WORK...UNDER "*NORMAL*" CONDITIONS
  (but not very forgiving!)
- SOURCE IS FAIRLY WELL DONE
  (not TAE, but ok!)
- HAS SOME HIGH LEVEL FUNCTIONS (MOREFUN)
  ...but not enough of 'em.
- INTER-PROCESSOR PROTOCOL SEEMS WELL DEFINED AND IMPLEMENTED
- IT SHOULD DO THE JOB WE WANT IT TO
- IT CONTAINS A NICE SET OF UTILITIES
- APPLE DEVELOPMENT SEEMS TO BE CONTINUING
  (work on V2.5, but on a very low priority basis)

MACWORKSTATION EVALUATION

"BUGS & QUIRKS"

- ALMOST EVERY INTERFACE UNIT HAS THESE!
- NORMAL COMMAND MODE A CHALLENGE
  (no event generated on text input)
- MHI DESIGNED TO BE THE "CONTROL PROGRAM"
  (it will have to "run" TAE)
- MHI DESIGNED TO RUN SINGLE APPLICATION

  -FILL IN THE BLANKS PROGRAMMING
  -REPLACE "MHIEVENTS" UNIT
  (essentially a collection of event handlers)

- POSSIBLE I/O & AST CONFLICTS BETWEEN TWM AND MHI
- HIGH INTERDEPENDENCY AMONG FUNCTIONS
  (but not much documented other than in each function description)
  (e.g., L_AddRecord and L_Stabs)
- MHI LOOSLY COUPLED "GAGGLE OF FUNCTIONS"
- FUN WITH MULTIPLE PROGRAMMING STYLES

  - TAE 'C' =
  - MHI 'C' =
  - LISA PASCAL = > SCHIZOPHRENIA
  - VMS OS =
  - MAC OS =

- SEE "LIST OF ENHANCEMENTS"

TAE CONVERSION EXPERIENCES FROM VMS TO UNIX ON A SUN WORKSTATION

Donald L. Anderson
Geology Department
Arizona State University

Sixth Annual TAE User's Conference
October 9, 1986

PRECEDING PAGE BLANK NOT FILMED

Outline

- Objective
- Computer system
- Executive conversion
- Applications programs
- Future work

OBJECTIVE

To produce a portable, stand-alone microcomputer-based image processing system to run under the Transportable Application Executive using the UNIX operating system.

COMPUTER SYSTEM HARDWARE

- Sun Microsystems 3/160C
- 68020 CPU (2 MIPS)
- 4 MB main memory
- Floating point accelerator
- 19" color monitor
- Keyboard and mouse
- 2 71MByte disk drives
- 1/4" cartridge tape drive
- 1152 x 900 x 8-bit resolution
- 8 x 24 color look-up table

COMPUTER SYSTEM CHARACTERISTICS

- Vanilla UNIX workstation implementation
- Multiwindow environment
- Built-in display processor

COMPUTER SYSTEM SOFTWARE

- UNIX O/S (BSD 4.3 and AT&T System V)
- Languages: FORTRAN 77, C, Pascal
- Make - program maintenance utility
- Source Code Control System (SCCS)

VICAR2 EXECUTIVE CONVERSION OUTLINE

- Use Goddard UNIX version of TAE
- Convert VICAR2 parameter I/O
- Convert VICAR2 file I/O
- Convert VICAR2 label I/O
- Place under SCCS

VICAR2 EXECUTIVE SOURCE CODE

- *Source Code*
  - File and label I/O: 2,300 lines of C
  - Parameter I/O: 500 lines of C
  - Support routines: 12,000 lines of C
  - Assembly: 150 lines -> 150 lines of C

- *VMS-UNIX differences*
  - System calls
  - Minor differences in C compilers
  - Assembly language -> C

VICAR2 APPLICATION PROGRAM CONVERSION OUTLINE

- Convert VMS FORTRAN to F77
- Debug programs
- Place under SCCS

VICAR2 APPLICATION PROGRAM SOURCE CODE

- *Source Code*
  - ~ 200 application programs
  - F77 preprocessor: 1700 lines of Mlisp
- *VMS FORTRAN-F77 differences*
  - Variable initialization
  - Change data types
    - BYTE -> CHARACTER*1
    - LOGICAL*1 -> LOGICAL
  - DO-ENDDO -> DO-CONTINUE
  - DO-WHILE -> IF-ENDIF
  - "!" (comments) -> "C"
  - Change HEX and OCTAL constant formats
- *Other Problems*
  - EQUIVALENCE statements
  - DEC data byte-swap order
  - Passing variable number of arguments

FUTURE WORK

- Rewrite F77 preprocessor in LEX/YACC.
- Select a display processor interface and convert to UNIX (if necessary).
- Convert appropriate subset of programs.
- Interface CD-ROM into system.

APPLICATIONS RUNNING UNDER TAE

N87
23161
UNCLAS

N87-23161

SOT, A RAPID PROTOTYPE USING TAE WINDOWS

Mark Stephens/GSFC David Eike/Carlow Assoc.
Elfrieda Harris/SAR Dana Miller/GSFC

PRECEDING PAGE BLANK NOT FILMED

C-2 *

SOT, A RAPID PROTOTYPE USING TAE WINDOWS

Mark Stephens/GSFC David Eike/Carlow Assoc.

Elfrieda Harris/SAR Dana Miller/GSFC

Mark Stephens, NASA/GSFC code 522, Greenbelt, MD 20771 301 344-5994

The window interface extension is a new feature of TAE. We are using this feature to prototype a Space Station payload interface in order to demonstrate and assess the benefits of using windows on a bit mapped display. We also want to convey the concept of Telescience, the control and operation of Space Station payloads from sites remote from NASA. This effort is part of a Goddard Space Flight Center (GSFC) research topic (RTOP) directed at identifying and evaluating user interface technologies applicable to system design in the Space Station era.

The prototype version of the TAE with windows operates on a DEC VAXstation 100. This workstation has a high resolution 19" bit mapped display, a keyboard and a three-button mouse. The VAXstation 100 is not a stand-alone workstation, but is controlled by software executing on a VAX/8600. Function calls can be made to this software to create windows and mouse sensitive regions on the VAXstation screen.

When TAE is executed, it uses these features to create it's own (TAE) window to display standard menu, tutor and help screens plus a status window which typically appears at the top of the screen (see figure 1). With the exception of beging displayed in a window, the contents of the TAE window are no different from what is normally seen on a standard video terminal. The status window allows the end user to bring down TAE, bring the current window to the top of the screen, store and recall windows and send an attention signal to all windows.

TAE also provides the user and the application programmer with access to the windowing features via C language functions and TAE commands. For example, to create a window, a programmer may use the TAE command WINDOW-CREATE or the C language function w_create. Other C language functions can be called to manipulate windows, for example, moving and storing windows for later recall. Another service is the creation of mouse sensitive rectangles from a C program With them, an application programmer can make 'mouseable' buttons which can report the results of a click back to the program.

Telescience will allow a user to control instruments associated with the payload from his or her home institution. The payload may be attached to the Space Station itself, or located on a co-orbiting or polar-orbiting platform. The user employs a workstation to send commands to and receive data from the payload. For the most part, this will occur without problems and the underlying NASA systems will be transparent to the user. However, there will be many payloads attached to the platform, each contending for a limited

number of resources and the ability to impact the environment of the platform. When conflicts arise for these resources and abilities, a user must interact with the Telescience system in order to them.

In order to demonstrate windowing and Telescience concepts, a payload was chosen for which a short scenario was developed. The payload we used as the model for the prototype is the Solar Optical Telescope (SOT). The SOT contains a primary mirror which can be moved to scan the entire solar disk. It can be directed to move to one of 254 predetermined regions on the sun and then to move in fine adjustments of a few arc-seconds. The light from the mirror is directed to a charged coupled display (CCD) imaging device and to one of three instruments. The instrument the scenario deals with is a scanning spectrograph known as the slitjaw. The slitjaw may be positioned by rotating the whole instrument assembly; it is desirable to do so when the slitjaw is not perpendicular to the feature being studied.

The scenario starts out with an end user sitting in front of the VAXstation 100. Adjacent to the VAXstation is an image analysis terminal used to display the CCD images. This analysis terminal is controlled via the VAXstation and is used in the target acquisition process as well as for the display of data. Prior to this session, the end user has already scheduled with NASA a block of time in which the user may perform real time (or 'near' real time) 'command and control of the SOT.

The prototype scenario is shown in figures 1 through 7. Figure 1 shows the VAXstation screen when the user first logs on to TAE. The root menu is displayed in the TAE main window. From this menu, the user chooses item number one to display a window containing status which is continually updated to reflect the current state of the SOT. The user also chooses item number two to display a list of experiments in another window. Figure 2 depicts a tutor for the date of this list just after the run command has been entered.

Again from the main menu the user begins target acquisition by selecting item number three to show the target acquisition menu (figure 3). By choosing item one, the end user is tutored for the region number. The user supplies the number experiment B seen in the list window. By 'running' the tutor, a command is sent to the SOT to move the mirror to the desired location. A CCD image is then sent down from the SOT and displayed on the adjacent image analysis terminal. The user then positions the phenomena of interest, a filament, in the center of the mirror's field of view by selecting item two from the target acquisition menu and supplying the X and Y coordinates of the resulting tutor. Again, running the tutor causes a command to be sent to the payload. For each action performed by the user, the status window is updated to reflect the position of the mirror.

On looking at the image analysis terminal, the user notes that the filament is not aligned perpendicularly with the overlaying image of the slitjaw. From item three of the target acquisition menu, the user attempts to rotate the slitjaw by supplying the

subsequent tutor with an angle. The underlying Telescience system informs the user via a dialogue box (figure 4) that torque on the platform will result from the rotation and the user has not scheduled an event which can create torque. The dialogue box presents the user with the choice of either taking the data from the slitjaw in its current position or finding the next time when the rotation may be performed. The user must respond to the box and does so by moving the mouse pointer over the 'Find Next' region and clicking the mouse button. The Telescience system will then inform the user at a later time when the rotation may be scheduled.

While the NASA system is processing the above request, the end user backs up to the main menu from which the user chooses item four. The user is in the process of filling in the resulting instrument configuration tutor when the system displays another dialogue box asking the user to select a time when the roll may be performed. The user may ignore the box for a while to complete the tutor or, as is the case here, respond immediately by clicking a mouse button with the pointer over the desired time region.

Figure 6 shows the screen at a slightly later time with the system informing the user that the rotation may now be performed. The user clicks on "OK", finishes with the instrument configuration tutor and returns to the rotate tutor. This time the execution of the rotate proc is successful and the resulting payload command positions the slitjaw correctly over the filament. The end user may now proceed to analyze the resulting data being collected by the slitjaw.

One of the main benefits of using windows is the way that multiple views into a system may be displayed on the screen simultaneously. For example, once the status window is displayed the end user can tell the well being and configuration of the SOT on a continual basis, without having to ask for the information. Another benefit is that messages can be displayed so as not to interrupt the current activity. The end user may respond to a message at leisure. Of course some messages must interrupt the user as was done when the slitjaw rotation could not be performed. This is done only when the system cannot proceed without information from the end user.

The VAXstation 100 TAE window extensions are themselves a prototype of a much more expansive system which will be designed in the near future. The SOT payload operations prototype will be used to help direct this effort. One exciting extension is the use of graphical interaction techniques, for example, performing target acquisition by tying the movements of the mouse to cross hairs over the CCD image. A click of a mouse button will set up a TAE proc to move to that position - it's another way of filling in a tutor screen! Some of our latest prototyping efforts at Goddard show such an interface but these efforts do not use TAE. We hope in the future to mesh these interface techniques with TAE to produce a highly interactive and easy to use interface.

Figure 1

ORIGINAL PAGE IS
OF POOR QUALITY

ORIGINAL PAGE IS
OF POOR QUALITY

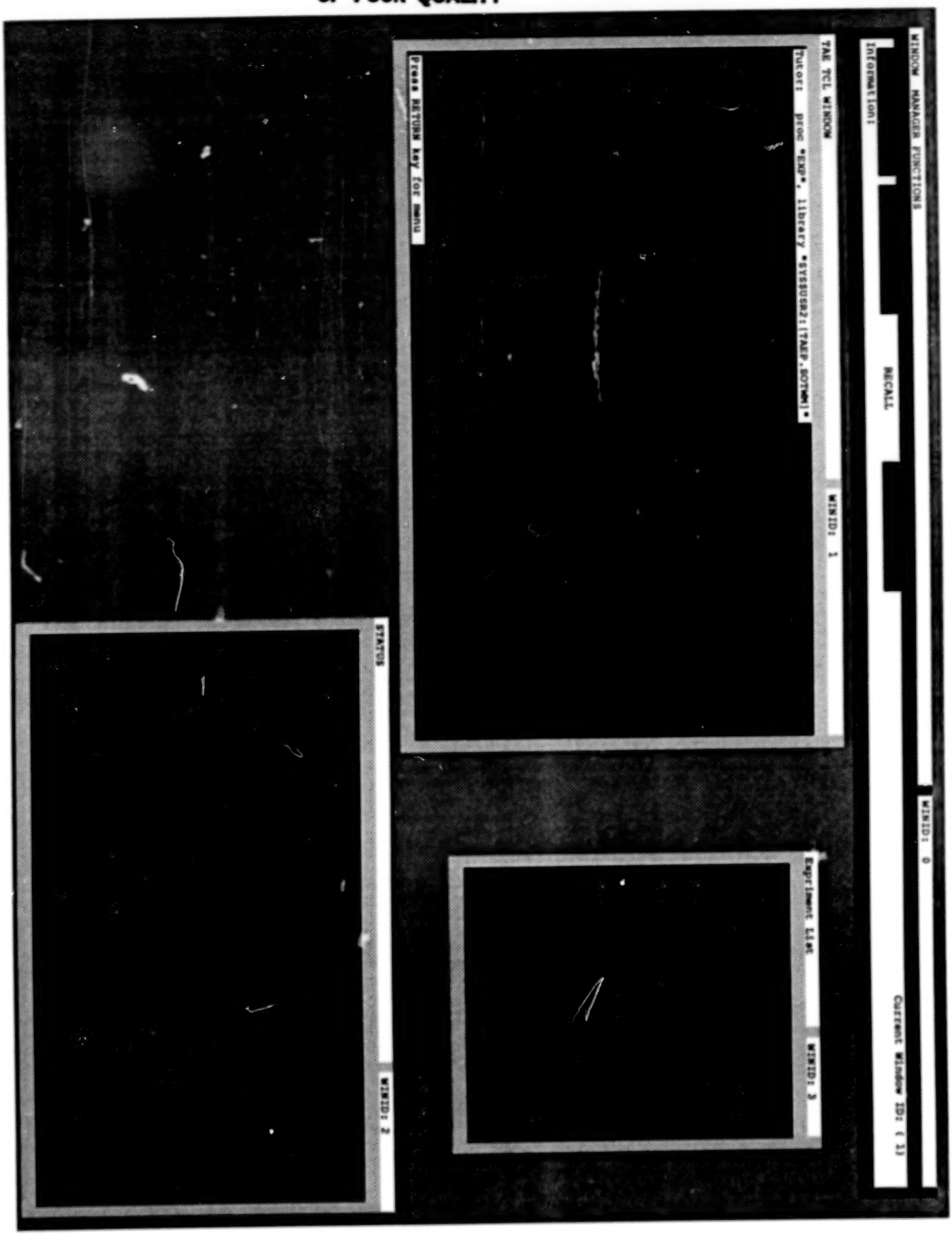


Figure 2

ORIGINAL PAGE IS
OF POOR QUALITY

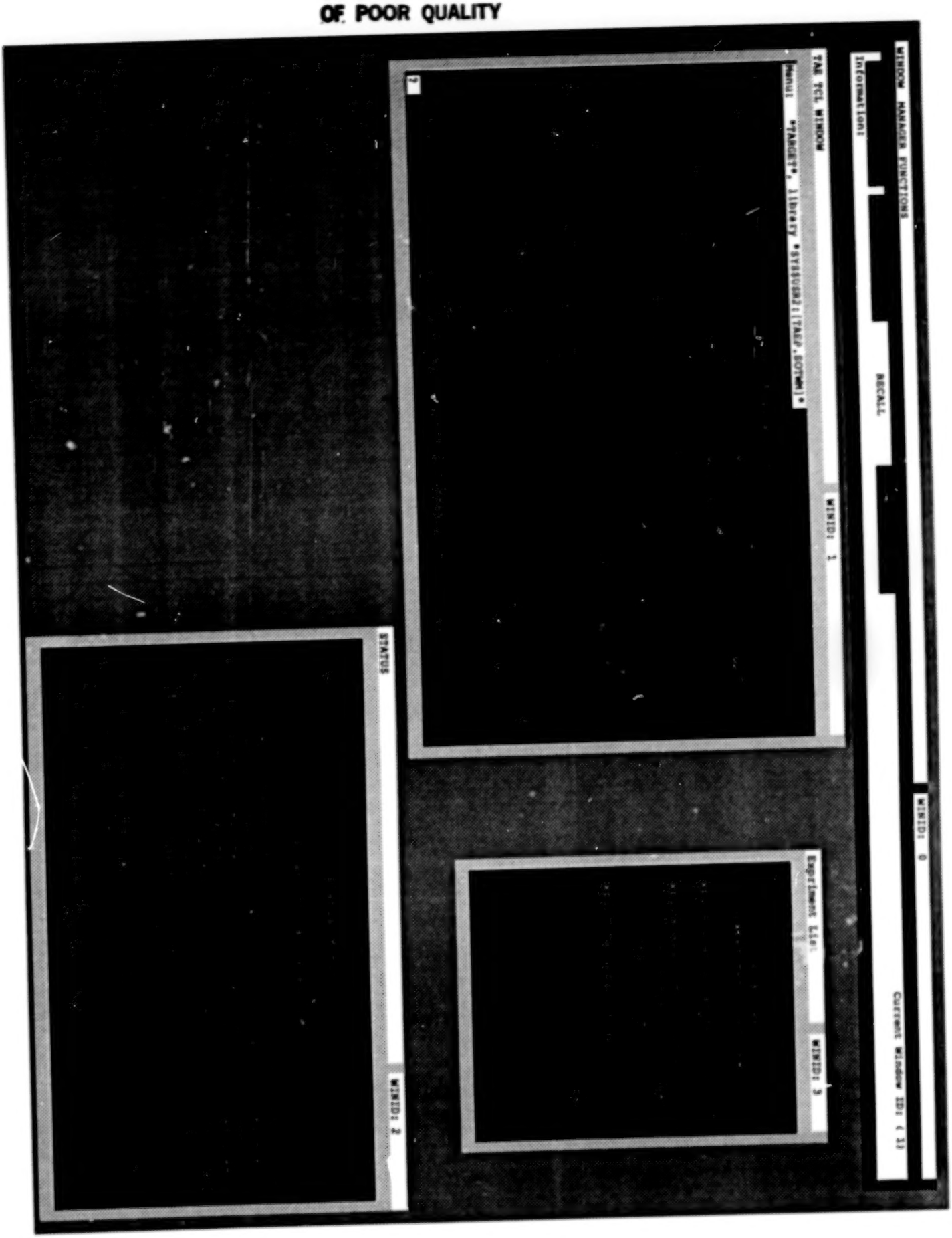


Figure 3

ORIGINAL PAGE IS
OF POOR QUALITY

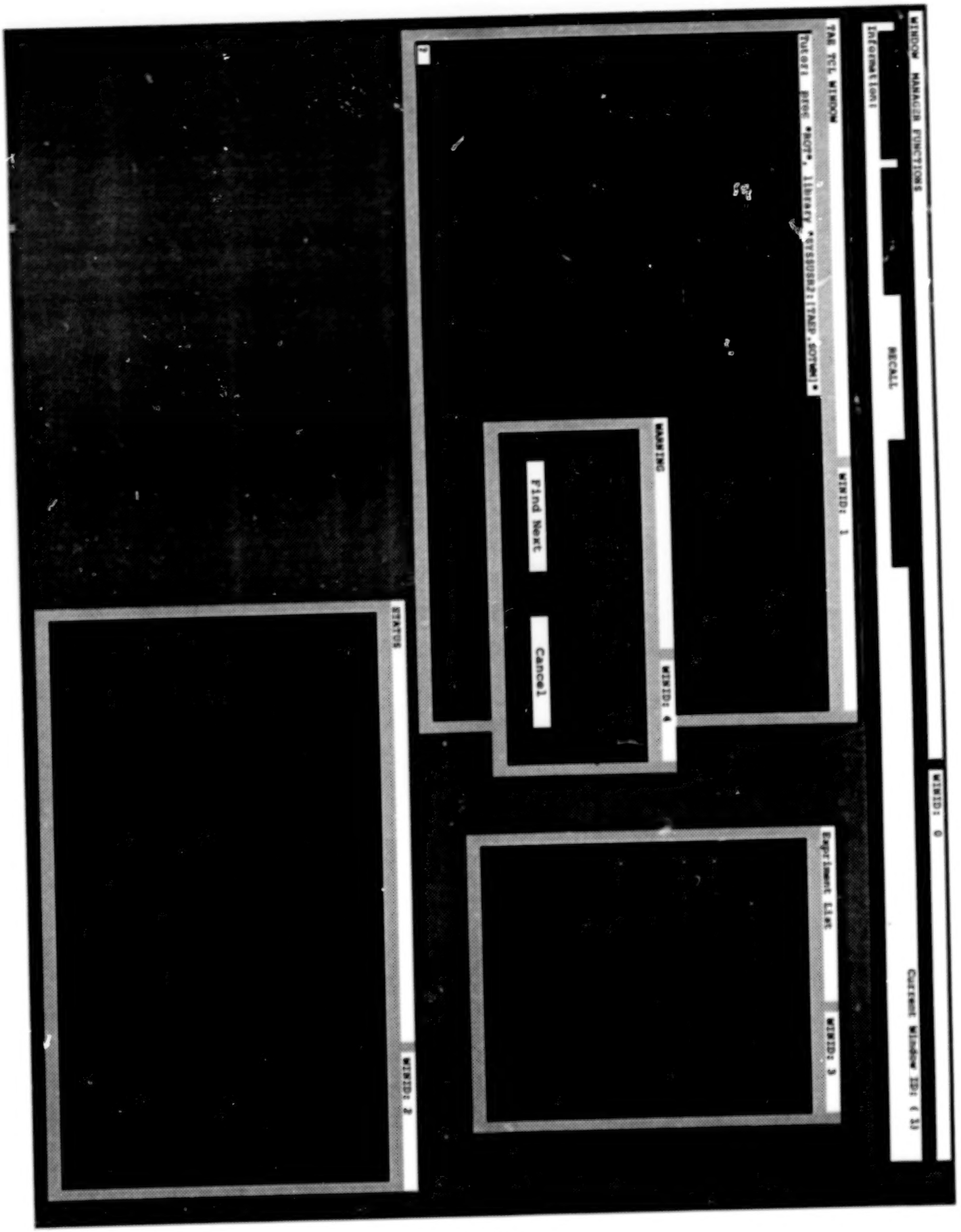


Figure 4

ORIGINAL PAGE IS
OF POOR QUALITY

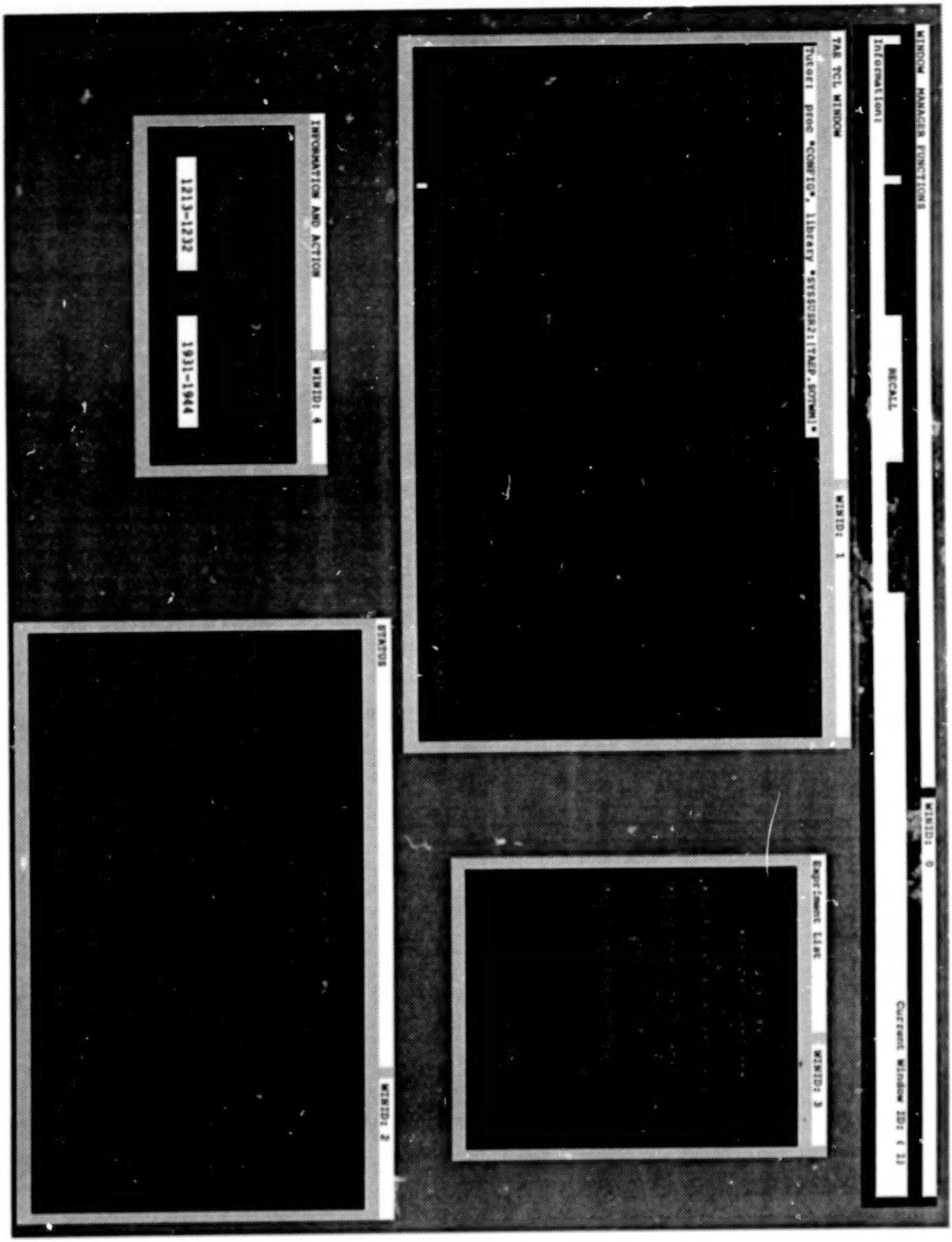


Figure 5

ORIGINAL PAGE IS
OF POOR QUALITY

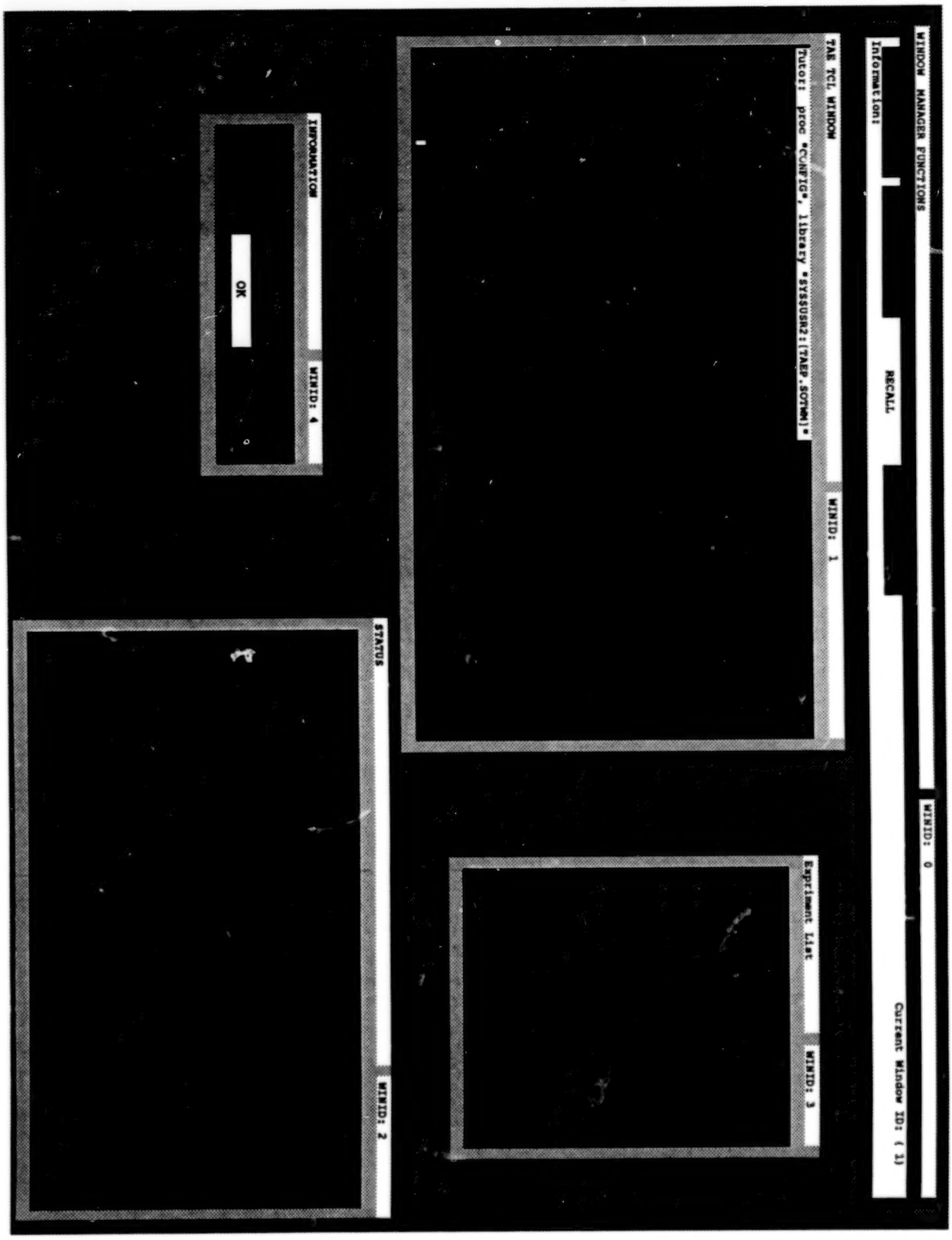


Figure 6

ORIGINAL PAGE IS
OF POOR QUALITY

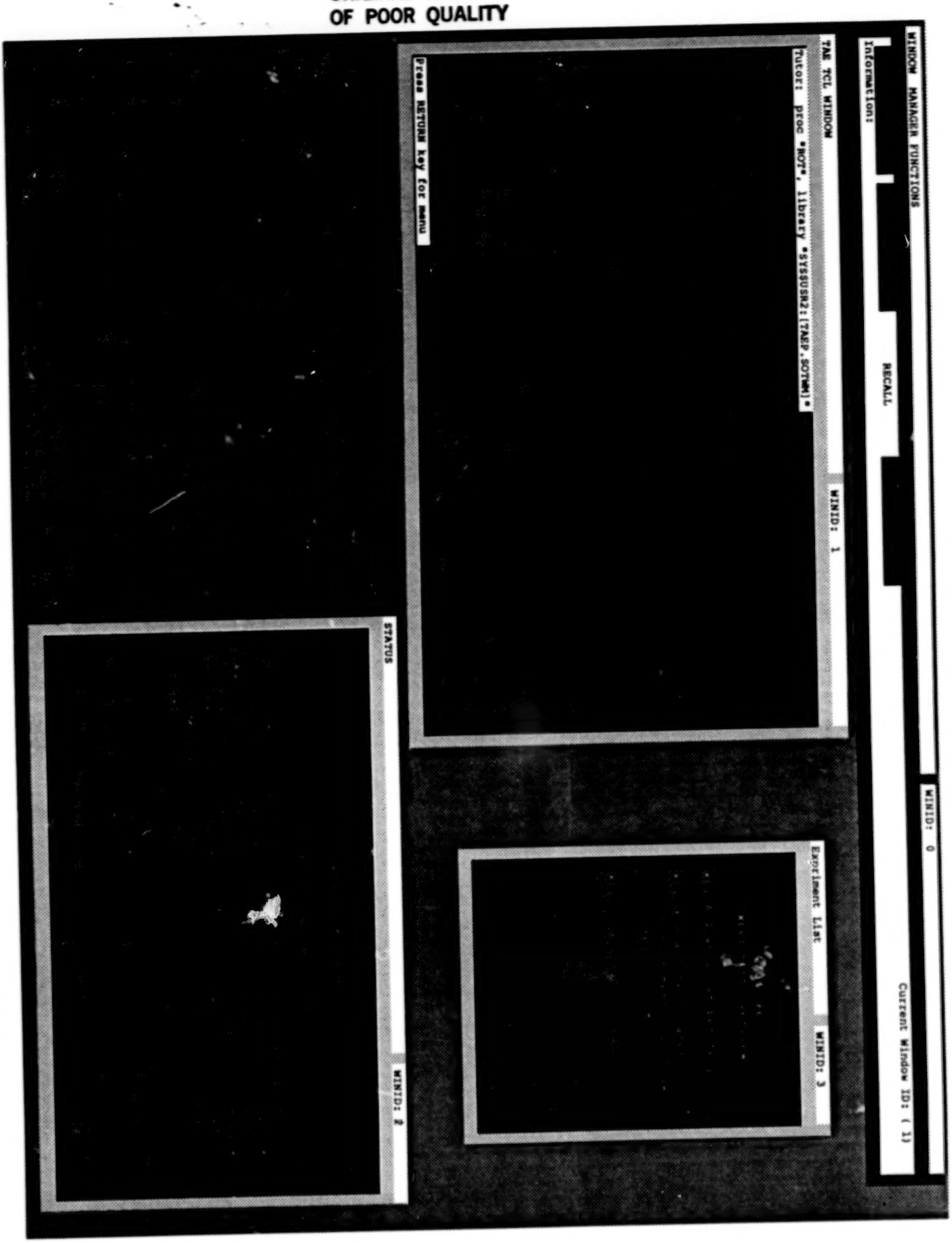


Figure 7

N87
23162
UNCLAS

N87-23162

2GCHAS–A HIGH PRODUCTIVITY SOFTWARE DEVELOPMENT ENVIRONMENT

Larry Babb
Computer Sciences Corporation
Systems Sciences Division

2GCHAS - A High Productivity Software Development Environment

Larry Babb
Computer Sciences Corporation
System Sciences Division

To the user, the most visible feature of TAE is its very powerful user interface. To the programmer, TAE's user interface, proc concept, standardized interface definitions, and hierarchy search provide a set of tools for rapidly prototyping or developing production software. The 2GCHAS (pronounced TWO GEE CHARLIE, Second Generation Comprehensive Helicopter Analysis System) project has extended and enhanced these mechanisms, creating a powerful and high productivity programming environment where 2GCHAS' development environment is 2GCHAS itself and where a sustained rate for certified, documented, and tested software above 30 delivered source instructions per programmer day has been achieved. The 2GCHAS environment is not limited to helicopter analysis, but is applicable to other disciplines where software development is important.

BACKGROUND

Predicting the characteristics and performance of helicopters is not a mature discipline; the theory is still developing and the computational tools based on the immature discipline are relatively undeveloped. The 2GCHAS project was established by the U.S. Army to develop a system which can predict the flight characteristics of a helicopter from a physical description of the vehicle. The objectives pertinent to the discussion in this article include;

1. Providing a standard set of helicopter analysis tools based on current theory,

2. Providing an environment which can assist developing new computational tools,

3. Providing the computational framework into which new or modified analysis tools can be inserted.

Three major considerations have driven 2GCHAS' design. The first is the requirement that 2GCHAS be operating system independent. With a rich set of functional requirements and initial implementations on VAX/VMS and IBM's MVS operating systems, 'pragmatic considerations have replaced operating system independence with operating system transportability. To achieve transportability, the user and programmatic interfaces have been defined to be constant across operating systems. Host dependencies have been restricted to the smallest number of units -- operating system procedures, 2GCHAS procedures, and software -- possible. The result is that most of 2GCHAS is and will be operating system

independent.

The second major consideration is that both end users and developers of new 2GCHAS analyses are helicopter engineers and scientists, not programmers or computer scientists. Their language of choice is FORTRAN. Their interest is in developing new analysis tools, modifying existing analysis tools, and using those tools to predict helicopter performance in a user friendly environment. 2GCHAS chose TAE to meet the user friendliness requirements.

The third major consideration is that every 2GCHAS Module (analogous to a TAE Process) has to be invokable both directly by the user and by other Modules and must be replacable at run time. Meeting the run time replacability requirement turned out to be the major contributor to the productivity of the 2GCHAS environment. As background to further discussion, the top level organization of 2GCHAS and the rationale for the replacabilty requirement and its implementaion (run time linking) will be described in some detail.

TOP LEVEL 2GCHAS ORGANIZATION

2GCHAS is divided into the Executive complex and tne Technology complex. The Technology Complex will consist of a large number of FORTRAN 77 Modules which perform the helicopter analysis. The Executive Complex provides services (e.g., user interface, data management, Module substitution) required by the Technology Modules and isolates them from the host operating system. When 2GCHAS is transported to a new operating system, only the Executive Complex is required to change.

MODULE REPLACABILITY AND RUN TIME LINKING

The replacability requirement for 2GCHAS Modules derives from the following considerations:

1. There will be a large number of Technology Modules in 2GCHAS. By design, as new helicopter analysis techniques are developed, Modules will be added to or modified within 2GCHAS.

2. A typical analysis will use relatively few of the available Technology Modules. Available Modules might supply alternative approximation methods, apply different flight simulation techniques, or compute different outputs.

3. It is difficult to predict in advance which Modules are required for an analysis. Modules are called as required by a combination of processing options, output results desired, and the description of the helicopter and its flight conditions.

ORIGINAL PAGE IS
OF POOR QUALITY

4. Any Module linked into an executable image uses resources even though it is never executed. For example, all Modules in an executable image are assigned virtual memory when the image is run. Modules which are not executed still consume virtual memory quota.

5. Multiple developers from different organizations will be developing new Modules throughout 2GCHAS' life. To provide a framework for developing new analysis techniques, the system structure must be quite open. Maintaining and distributing a standard set of object Modules is difficult for an open system.

The solution to the replacability and transportability requirements is to have run time linking, to have a Module call look exactly like a subroutine call, and to define a Module to be a FORTRAN subroutine. There are only four minor differences between a Module and any other FORTRAN subroutine:

1. The last argument is the completion status of the Module;

2. The ENTRY statement is not permitted;

3. COMMON is not permitted for inter Module communication;

4. A special set of comments, the Preamble, is required to provide information about the Module and its arguments.

Suppose, for example, Module A calls Module B. Each Module is compiled, producing an object file. The object files are then linked by the Linker into an executable image which can be executed using the RUN command. (There must be a main program linked with the subroutines, but it has been omitted here to show the Module linkage process more clearly.) Figure 1 shows how the Modules are linked using the standard linking procedure.

Figure 2 shows the run time linking technique used by 2GCHAS. From the Module source, DEFMOD (DEFine MODule) produces a Module caller source file and then compiles both source files to produce object files. In this example, Module B is defined first. DEFMOD produces the source file for B Caller from the Module B source file, then compiles both source files to produce object files for B Caller and the body of Module B. The object file for B Caller is put in an object library containing the Module callers for all defined Modules. The object file for the body of B is linked to produce a shareable image of B. Similarly, when Module A is defined, the object file for A Caller is put in the Module caller object library and the object file for the body of A is linked with the library of Module callers to produce a shareable image for Module A. Since Module A calls Module B, B Caller is linked into the shareable image for Module A.

ORIGINAL PAGE IS
OF POOR QUALITY

When Module A executes the "CALL B" statement, it actually executes a subroutine call on B Caller which has been linked into the image in place of Module B. The B Caller subroutine, created by DEFMOD, calls an Executive service, Module Execution Control (XMEC) to activate the shareable image for Module B. When XMEC is called, it determines if the image has already been activated. If the image has not been activated, XMEC calls an operating system service to activate it. After XMEC activates the shareable image of Module B (or finds it already activated), XMEC executes a subroutine call on Module B and passes the argument list from the subroutine call executed by Module A. When Module B completes its execution, it returns to XMEC, which returns to B Caller, which returns to Module A. On VAX/VMS systems, XMEC uses the Library Service LIB$FIND_IMAGE_SYMBOL to activate shareable images.

From Module A's point of view, a standard subroutine call has been executed. From 2GCHAS' point of view, the Module is assigned virtual memory and other resources only if it is executed.

Run time linking, like all solutions to difficult problems, contains tradeoffs. The advantages of run time linking are;

1. Virtual memory and other resources are allocated to Modules only if the Modules are executed.

2. Enhancements to Modules can be tested by Module substitution at run time.

3. The option of linking a Module directly using the standard Linker remains available with no change in the Module source, since the source is standard FORTRAN 77.

The disadvantages of run time linking are;

1. Program execution time is increased. The first time a Module is called on a VAX 11/785, an elapsed time of about 0.2 seconds is required to activate the Module. Subsequent calls on the Module take considerably less time, although more than subroutine call. Because helicopter analysis runs will be computationally intensive, the overhead time required to activate the analysis Modules will be a small portion of the total job.

2. FORTRAN COMMON blocks cannot be used to communicate between Modules linked at run time. However, subroutines which are contained within a single Module may communicate with each other through COMMON blocks as usual. In 2GCHAS, the Executive contains data management services which provide a data structure intended to replace FORTRAN COMMON blocks in communicating between Modules linked at run time.

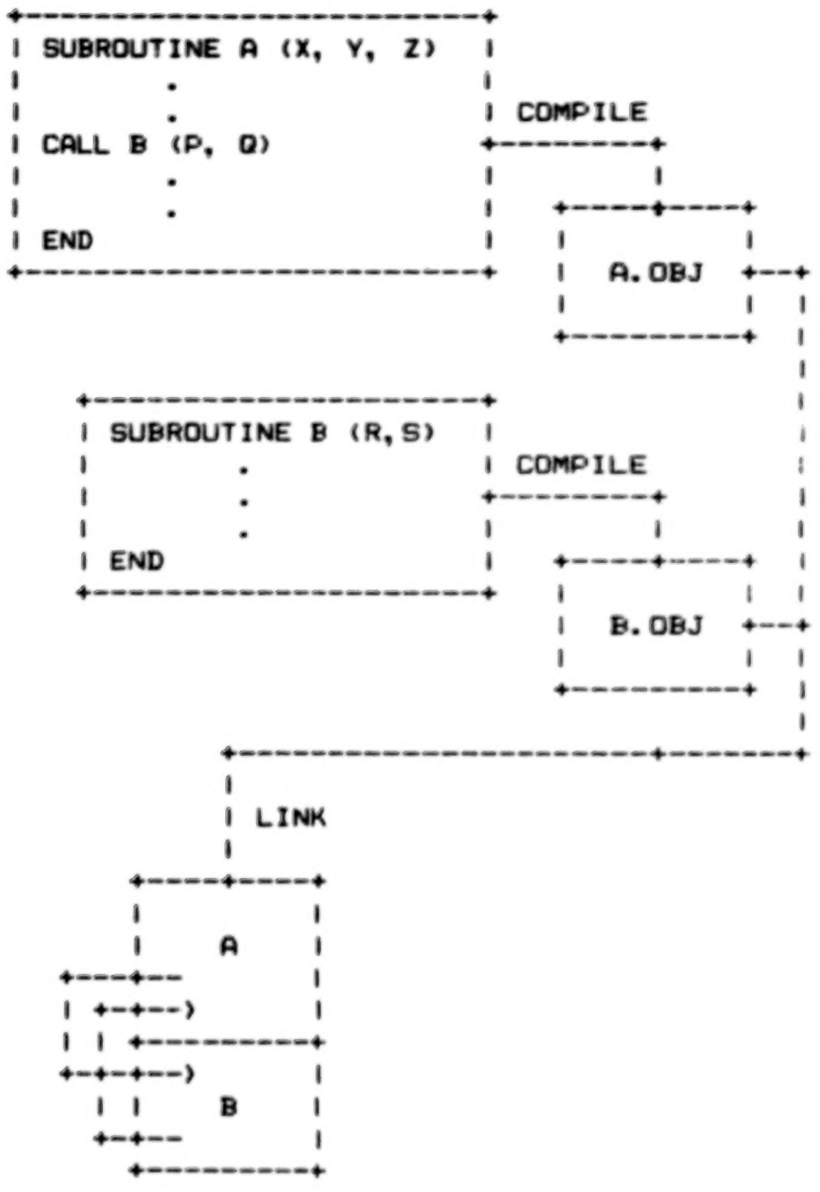


Figure 1. Standard FORTRAN Subroutine Linkage

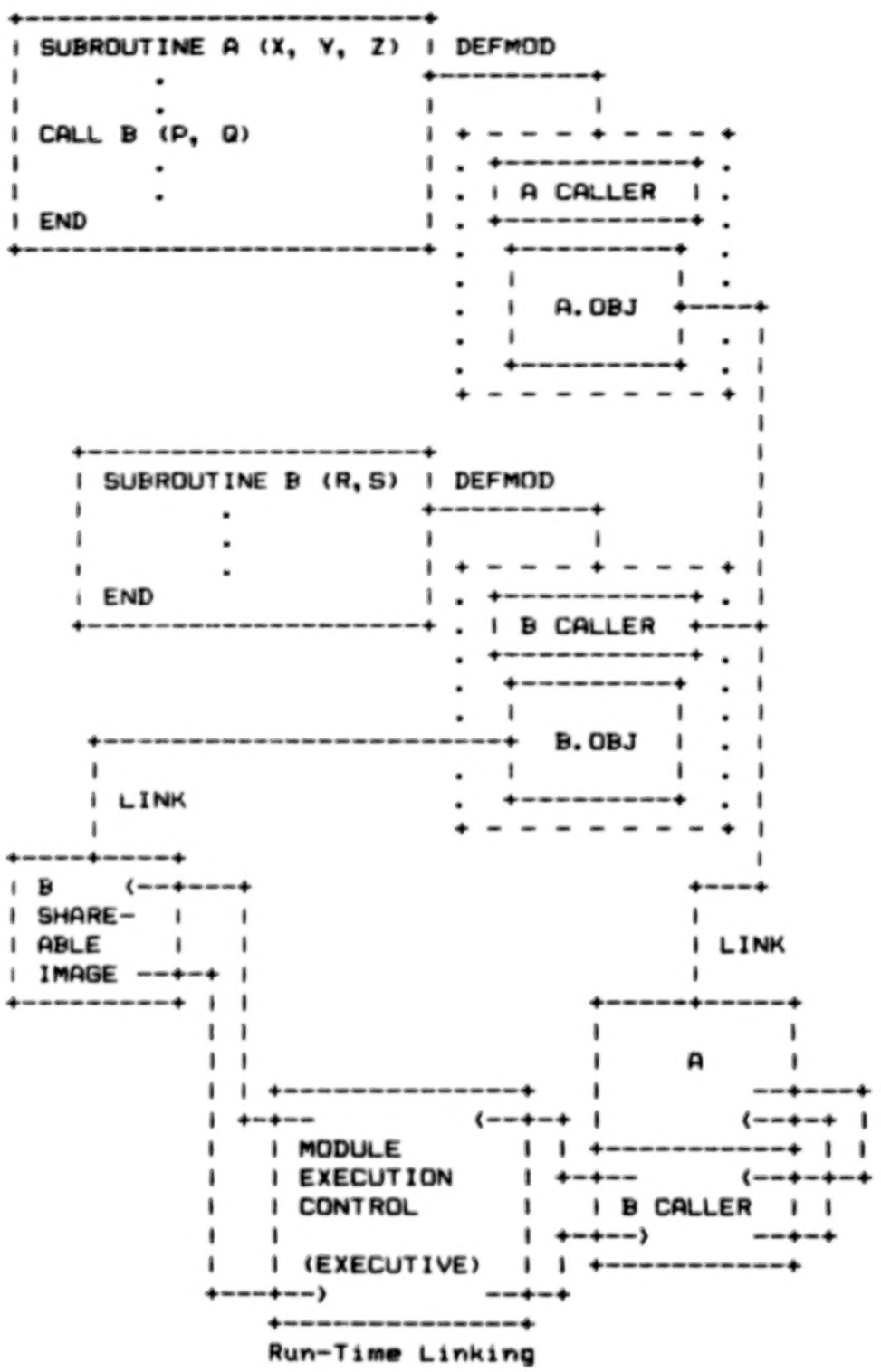


Figure 2. 2GCHAS Module Run Time Linking

DEFMOD TAKES THE DRUDGERY OUT OF MODULE CREATION

DEFMOD's role in creating Modules for run time linking has already been described. DEFMOD provides another feature for developing and unit testing FORTRAN subroutines. From the Module preamble, DEFMOD creates the process PDF and all of the VBLOCK references necessary for the user to invoke the Module directly.

To appreciate the amount of effort that DEFMOD saves, consider a TAE process. It consists of at least two separate files -- the process itself and the process PDF. In the process PDF, information about each parameter can be in as many as three disjoint places -- the PARM statement, the LEVEL1 description, and the LEVEL2 description. The information in each place within the process PDF must be consistent. The process PDF, in turn, must be consistent with the VBLOCK references in the process itself. Consistency of physicaly separated information is hard to achieve and the requirement for it can lead to increased development and maintenance costs. In the case where the help text is separate from the process PDF, there are three files which must be consistent.

In contrast, a 2GCHAS Module source file contains all of the information needed by DEFMOD (DEFine MODule) to create the components necessary to execute the Module. The Module Preamble, that special set of comments at the beginning of a Module, contains all of the information about the Module and its parameters. The preamble format rules are less restrictive than the rules for a process PDF. The parameter information which becomes the PARM statement, the TAE LEVEL1 text, and the LEVEL2 text is contiguous, not separated. The input which becomes the TAE LEVEL1 parameter information is broken automatically (and reasonably) to fit into the 32 column width restriction.

To be invoked directly by the user, the Module must have, in addition to a process PDF, VBLOCK references for the parameters. But a Module is a FORTRAN subroutine, it contains no VBLOCK references. And the Module's parameters can be of any FORTRAN data type (i.e., INTEGER, REAL, CHARACTER, DOUBLE PRECISION, COMPLEX, or LOGICAL) not just REAL, INTEGER, and STRING.

From the preamble in the Module source file, DEFMOD constructs the necessary files. The process PDF contains the parameter definitions and HELP information. DEFMOD constructs a FORTRAN program (called the Module Main) which contains the VBLOCK references, the necessary conversion code to go between TAE data types and the larger set of FORTRAN data types, and a FORTRAN CALL to invoke the Module.

LOGICAL parameters are an especially good example of the processing provided by DEFMOD. The Module preamble states that the parameter is of type LOGICAL. In the process PDF, the parameter's type is (STRING, 1) and its valid values are the letters "T" and "F". In the Module main, the letter "T" from the input parameter is converted to a FORTRAN logical .TRUE.; the letter "F" to FALSE. The Module main CALLs the Module with a FORTRAN LOGICAL parameter. After successful return from the Module, LOGICAL .TRUE. is converted to "T" and .FALSE. to "F" if the parameter is OUT or INOUT (TAE parameter type NAME).

Creating a Module with DEFMOD saves much of the effort normally required to implement a TAE process and yields three major benefits. The first is the ease with which changes are made to the calling sequences of Modules. It is easy to add, delete, or change a parameter or the parameter's attributes by editing the preamble and FORTRAN statements. It is similarly easy to change the help information for the Module or for individual parameters.

The second benefit is the ease with which small pieces of software (e.g., finding the roots of a polynomial using Newton's method) can be quickly prototyped as Modules and tested from command or tutor mode. Although there is some effort required to put a preamble in a Module, that effort is small compared to the effort of either creating a test driver for the Module or for implementing VBLOCK calls.

The third benefit is that packaging decisions can occur very late in the development cycle. Because the preamble information consists of FORTRAN comments, code developed as a Module can remain a Module or can be made into a subroutine linked in to a larger Module. Such a packaging decision requires no change to the source. Thus, the decision to make a unit into a Module or not is not critical. Not only can the decision can be deferred, but it is easily changed if made incorrectly.

LOOSE COUPLING OF MODULES CONTRIBUTES TO PRODUCTIVITY

Edward Yourdon and Larry Constantine in Structured Design define coupling as the degree to which one software unit depends on knowledge of another software unit in performing its task. They argue that loose coupling leads to a design that is easier to develop and easier to maintain and that close coupling leads to a design which is more difficult to develop and maintain. One of the working definitions for a loosely coupled system is that the software units are black boxes; that is, the only information other software units know about the black box are its name, inputs, and outputs. Any blackbox unit can be replaced with a different unit having the same name, inputs, and outputs with no effect on the system.

2GCHAS Modules have three attributes that make them a loosely coupled system:

1. 2GCHAS Modules are FORTRAN subroutines with names and specific calling sequences; i.e., Modules are black box software units;

2. COMMON is not allowed as a communication mechanism between Modules, thus removing the greatest source of data coupling from 2GCHAS Modules;

3. Modules are linked at execution time, not at compile time or linkage edit time.

So how do Modules help improve productivity? The primary benefit is that any Module can be changed with little concern for other Modules in the system. As long as the new version of the Module has the same name and calling sequence, no changes are required elsewhere. For software development, top down implementation is easy. As they are developed, the functional Modules replace the limited function stub Modules. The replacement only requires that the functional Module be processed by DEFMOD.

The second benefit arises from the fact that Modules are located and loaded at run time via an extension of the TAE hierarchy search. A developer's personal version of any Module can be invoked at run time instead of the "official Module" in the 2GCHAS system library. Thus, a developer can have corrected versions or completely new versions of existing Modules in a private account and can test those Modules as part of a complete system. This means that new Modules to be integrated have been already been tested as part of a complete system.

CHANGE CONTROL HELPS PRODUCTIVITY

2GCHAS has created a set of configuration management support tools which enhance the productivity of all developers. These support tools provide formal change control, automatic system generation from source changes, and continuing operation of the previous version while a new system is being generated.

TAE is delivered for VAX/VMS systems, the System Manager's Guide describes how to set up the TAE version tree so that only the TAE system manager has write access. Change by anyone other than the system manager is effectively prevented. VAX/VMS file protection and the TAE hierarchy search provide both control and flexibility. 2GCHAS uses these basic TAE concepts.

It was recognized early that a mechanism to quickly and reliably introduce changes to 2GCHAS in a controlled manner was required. The mechanism, CHASGEN, consists of seven manual steps, each either a DCL procedure or a 2GCHAS procedure. The CHASGEN process is started when the Configuration Management officer is notified that new source units are being delivered. The Configuration Management officer begins the CHASGEN by copying the new source units -- not object files, libraries, executables, or message help file indices -- into the 2GCHAS version tree. Based on each source unit's file type and update date, the appropriate actions -- compile, library update, link, MSGBLD, etc. -- are performed. Within a short time, the Configuration Management officer, not a senior programmer, has constructed a new 2GCHAS Executive.

CHASGEN has proven to be quick and reliable. However, interruptions to users or developers is costly. Therefore, CHASGEN does not operate against the current version tree, but against a newly created version tree. The previous version remains untouched and operational. When the newly created version tree has been tested and proven to be good, it is then available for use.

STANDARD TAE FEATURES AS PRODUCTIVITY AIDS

The focus thus far has been on the 2GCHAS extensions to TAE and the benefits derived from those extensions. In large measure, however, those extensions have been possible within the real constraints of time and money because TAE provided such a stable platform upon which to build. This last section provides a brief summary and, in some cases, a recapitulation of the TAE features which have directly contributed to a high level of productivity within the 2GCHAS project.

For the programmer at a terminal, HELP and TUTOR, not paper documents, provide 2GCHAS documentation without work flow interruptions of referring to a manual. Fewer interruptions means that the programmer more effective uses time.

The hierarchy search and its extension to Modules provide a simple and easy mechanism to check out and test new software. Little effort is required to have a private version of some part of 2GCHAS and, so, there can be more effort available for other tasks.

TAE provides many features for tailoring the user interface. So far, every 2GCHAS developer has used ULOGON to tailor the user interface. There is a great deal of idiosyncratic tailoring, but two characteristics are common. The first is to use DEFCMD to make commonly used VAX/VMS DCL commands available inside 2GCHAS. The second is having PDFs which allow various editors to be invoked from inside 2GCHAS. The DEFCMDs and editor PDFs means that the 2GCHAS and DCL environments are the same in many respects, making the transition from one environment to the other smoother for the developer.

2GCHAS chose to use message help files as a primary means of providing online diagnostic information. The simple and powerful mechanisms of message help files and the MSGBLD utility contribute to productivity in three ways. The first is that messages and their corresponding help can be grouped. Having the help for possible messages produced by related software units -- a TAE facility -- together in a single place makes it easier to get consistency of information across messages. The second is that the 2GCHAS project experience indicates that message keys tend to get reused for similar or identical situations. When this happens, there is less help text to generate and the message text can be used to provide occurrance specific information. The third property of message files which acts to reduce effort is that they are located outside of the software units to which they apply. This means that the continual activity of making clarifications and additions to message help does not appect the software units themselves.

TAE features have significantly reduced the testing effort for 2GCHAS. Formal testing, the use of predefined and documented test scenarios generating output for comparison against expected results, is a 2GCHAS project requirement. The contents of each new system generation or build are quickly cataloguec with script files containing TUTOR and HELP commands for each 2GCHAS service. If the script completes without stopping, then all of the services are present. A missing service stops the script -- $MESSAGE is set to "ATTN" -- allowing the tester to log the missing service. The script files provide for a level of speed and repeatability that a person at a terminal with a written test scenario can only dream of approaching.

2GCHAS test procedures have been designed to be both self verifying and self reporting. Basically this means that the test determines whether or not it succeeded and then reports the success to the test conductor. The effort to perform and document formal testing is reduced because of these test procedures. In addition to scripts, tests are implemented as procedures and processes. $SFI, the success/fail indicator, is set by every test and reported in the session log. The STDOUT command qualifier is used to retain output generated by individual tests. Listings of session logs and test output provide most of the formal test documentation, further reducing the testing effort.

ORIGINAL PAGE IS
OF POOR QUALITY

2GCHAS - A High Productivity Software Development Environment

CURRENT 2GCHAS STATUS

The Executive Complex of 2GCHAS is being developed under contract by Computer Sciences Corporation (CSC) at Ames Research Center in a series of five builds. CSC delivered Build 3 of the Executive to the Army in September, 1986. Technology Module development began in early 1986. Technology Module developers will access Executive Build 3 (and Build 4 later) either by telecommunicating with AMES or by having a copy of the 2GCHAS on their own VAX. Run time Linking was included in Builds 1, 2, and 3 of the Executive; was tested by the Army as part of normal build testing; and was used by CSC for developing Builds 2 and 3.

SUMMARY

TAE provides both tools and concepts for achieving high productivity software development. Additional requirements have led the 2GCHAS project to extend TAE's tools and concepts with resulting additional increases in productivity.

ACKNOWLEDGEMENT

I would like to thank the staff of the 2GCHAS project for their support and for their diligence in creating the products and results reported here. Special thanks to Clark Oliphint for providing text and illustrations for the discussion of run time linking.

TECHNICAL PAPERS ON TAE SUBSYSTEMS

PRECEDING PAGE BLANK NOT FILMED

N87
23163
UNCLAS

N87-23163

AN OVERVIEW OF THE CATALOG MANAGER

Frederick M. Irani
Science Application Research
4400 Forbes Blvd.
Lanham, Maryland

PRECEDING PAGE BLANK NOT FILMED

AGENDA

Introduction
Brief History
Requirements
Naming Conventions
Catalog Structure
User Operation of CM
Fortran/C Callable Routines
File Attributes
Archive/Retrieve Capabilities
LAS User Scenario
Configuration
Performance and Future Enhancements

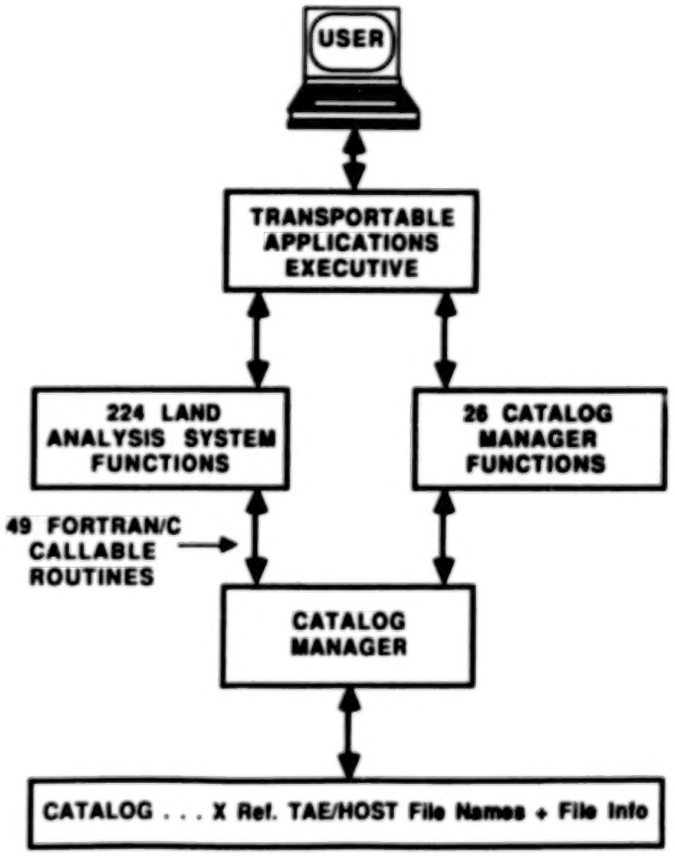
INTRODUCTION

The Catalog Manager (CM) is being used at the Goddard Space Flight Center in conjuction with the Land Analysis System (LAS) running under the Transportable Applications Executive (TAE). CM maintains a catalog of file names for all users of the LAS system. The catalog provides a cross-reference between TAE user file names and fully qualified host-file names. It also maintains information about the content and status of each file.

BRIEF HISTORY

- o Original Design for CM was borrowed from the VISSR Atmospheric Sounder (VAS) software in 1981.
  VISSR = VISIBLE INFRARED SPIN SCAN RADIOMETER
  - The VAS software was written by Computer Sciences Corporation for NASA.
  - CM under VAS was used primarily for it's file searching capabilities.
  - The this system met the overall requirements for an LAS Catalog Manager.
  - LAS has used a modified version of this software since 1981.
- o Has since undergone several internal design changes and enhancments.
- o It has been rewritten in several times in FORTRAN.
- o Has been maintained by the Century Corporation recently.
- o It is now written entirely in C.
- o Has been ported to run under the UNIX operating system as well as VAX/VMS.

An Overview of the Catalog Manager


USER
TRANSPORTABLE APPLICATIONS EXECUTIVE
224 LAND ANALYSIS SYSTEM FUNCTIONS
26 CATALOG MANAGER FUNCTIONS
49 FORTRAN/C CALLABLE ROUTINES
CATALOG MANAGER
CATALOG . . . X Ref. TAE/HOST File Names + File Info

REQUIREMENTS

o CM is designed to provide:

  o A standard naming convention across operating systems.

    - This is done as a complement to the user friendliness provided by TAE in LAS.

    - LAS users will be familiar with the naming convention regardless of the operating system of the host computer.

  o Storage of file attributes is provided.

    - The catalog stores information about each file such as the file type. That is, whether the file is an image, statistics file, look-up-table or some other type of file.

    - There are many attributes which may be stored for a file. These will be discussed later.

  o Associations between files.

    - Each catalog entry may contain a list of other file names which are closely associated with that entry.

    - For example, a CLASSIFIED image name may have an associated file list. This list may include the names of the raw image and statistics files that were used to create it.

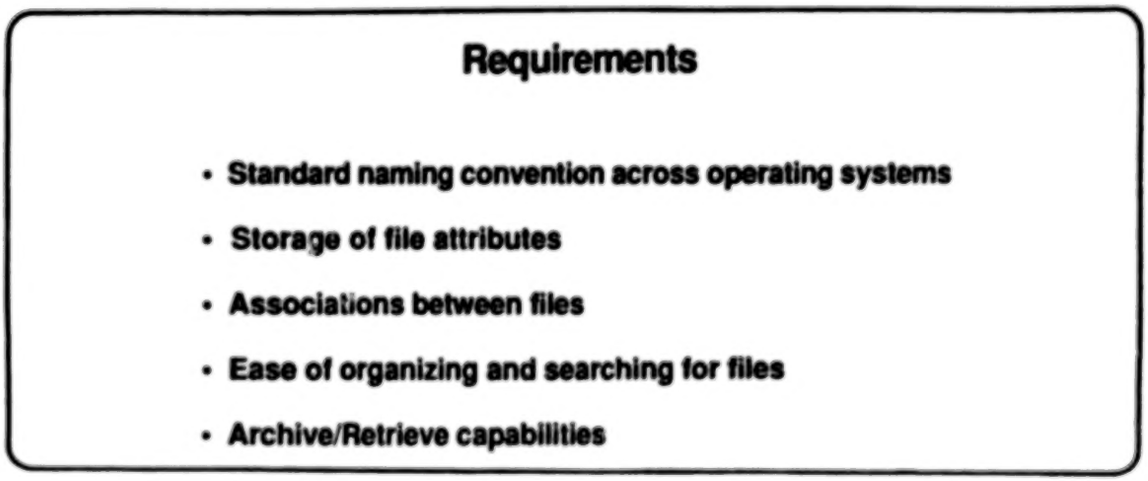
  o Ease of organizing and searching for files.

    - The structure of the naming convention and maintenence of file attributes allows users to manage their files efficiently.

  o Recently, Archive and Retrieve capabilities have been added to CM.

    - Functions have been added to be used at the programmer, user and system manager level to handle off-line files.

    - This feature will be discussed in greater detail.


Requirements
• Standard naming convention across operating systems
• Storage of file attributes
• Associations between files
• Ease of organizing and searching for files
• Archive/Retrieve capabilities

o NAMING CONVENTIONS

o The "TAE" name

- The actual contents of each file created in LAS resides in the user's default directory of the host computer.

- The name of this file is determined by LAS functions at the time it is created. This is called a host-file name.

- The LAS user never refers to host-file names however. Instead they use TAE names which follow the naming convention prescribed by catalog manager.

- The catalog then provides the cross-reference between the host-file and the TAE name.

- Host-files are automatically "cataloged" along with a TAE name by LAS functions. CM functions exist, however, to catalog a host-file with a TAE name without using LAS functions.

Structure:

o Root
  - The Top Root consists of the User's log in name.
  - Therefore, a top root name is required only to refer to another user's files.

o Directory Qualifiers

  - One to 20 qulifiers of up to 8 characters.
  - maximum file name length of 100 characters.

o File Version (Represents the leaves/endpoints of the catalog tree.)
  - Separated from directory name by a semicolon.
  - Up to 99 versions of a file are permitted.
  - "L" for Latest or "B" for Best version are permitted.

Wildcard Characters

- CM accepts an asterisk as a "wildcard" qualifier value.

- The wildcard allows users to specify several files without explicitly naming each.

- CM will process files which match all characters in a given file name regardless of the characters which occupy the position of a wild card.

- Only one wildcard is permitted per directory qualifier.

Default Root:

- Users often have files with several identical leading qualifiers.

- For convience the default root may be extended to include a desired directory name.

- This default root is temporary. It will exist only for the length of a single TAE session.

Alias Names:

- Used as a convienience to name files which are used frequently.
- Are permanently maintained in the catalog untill deleted.

CATALOG STRUCTURE

o The Catalog contains file names and associated information about the file only.

o Tree Structured Catalog

- User Roots (Catalog sub-tree) - point to directories.
- Directory/Branch names (nodes of tree) - point to file versions.
- File versions (leaves) - point to attributes and alias names.
- The tree structure of the catalog is strongly reflected by the naming convention used under TAE.

Naming Convention

STRUCTURE

 # ROOT. QUAL 1. QUAL 2... QUAL N; VERSION #

WILDCARDS

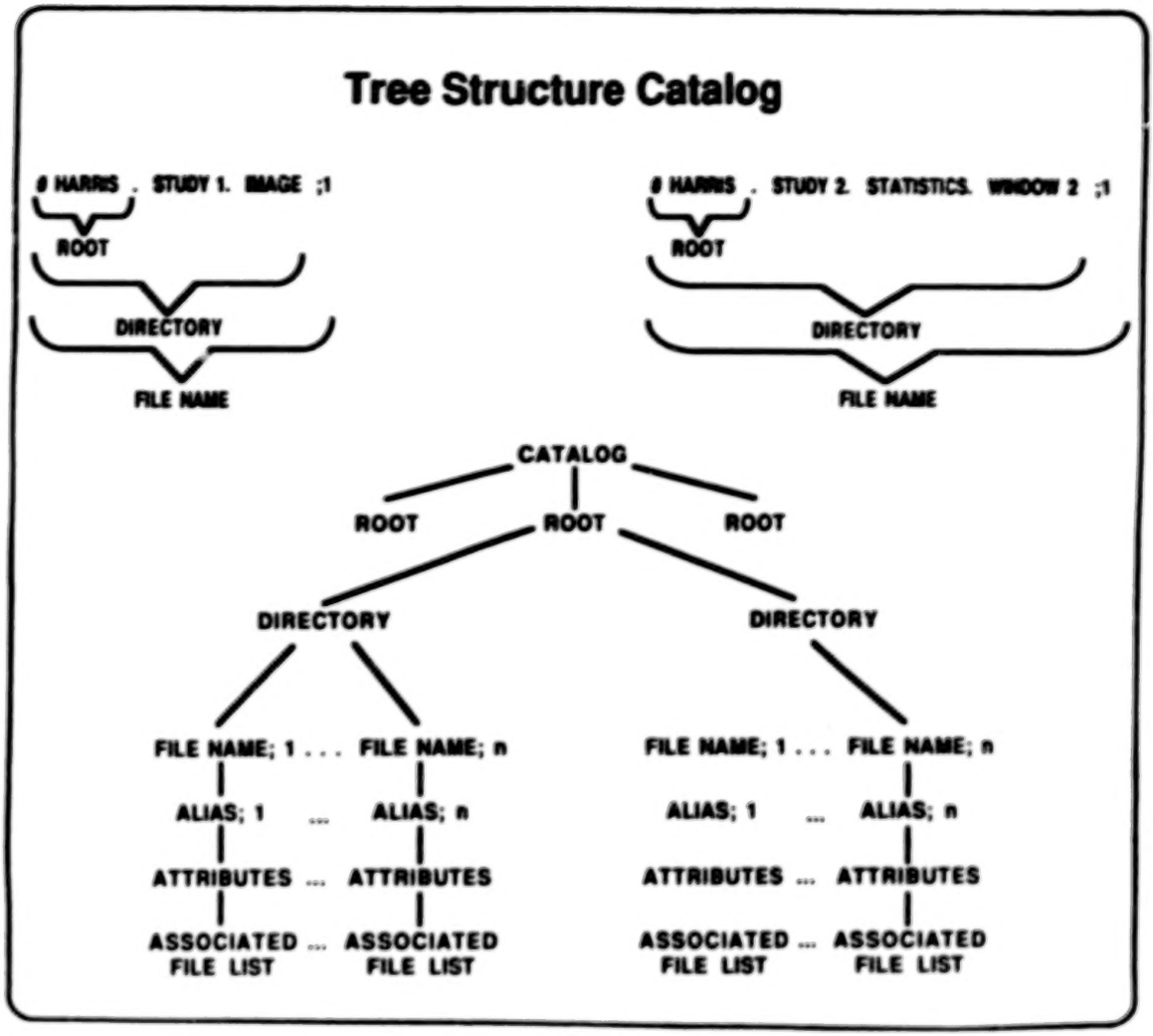
 # HARRIS. *. IMAGE*. = #HARRIS.STUDY1.IMAGE1
#HARRIS.STUDY3.IMAGES

DEFAULT ROOT

CMSET-ROOT #HARRIS. STUDY 2
STATISTICS = #HARRIS.STUDY2.STATISTICS

ALIAS NAMES **(2 ROOTS)**

CMALIAS-CREATE	**$ JANUARY**	**STUDY 2. STATISTICS**
	$ FEBURARY	**STUDY 1. IMAGE. STATISTICS**



USER OPERATION OF CM

o Users may run CM directly from a TAE command line or tutor screen.

o CM functions are available to handle

- File Naming
- Cataloging/Uncataloging Host-files
- Obtaining Host-files Name from TAE Name and Reverse
- Editing file attributes and Associating file names
- Searching the catalog by name and or attribute
- General operations: such as Setting default directories
- Archive / Retrieve capabilities

Catalog Manager User Operations

- CMALIAS-CREATE - Assign an alias.
- CMALIAS-DELETE - Deassign an alias from a directory name.
- CMALIAS-LIST - List aliases.
- CMALIAS-RENAME - Change an alias.
- CMARCH - Catalog files to tape.
- CMCAT - Catalog a host file.
- CMCHGATR - Change cataloged file attributes.
- CMDEL - Delete a cataloged file or a host file.
- CMLIST-ATTR - List attributes.
- CMLIST-DIR - List file in a directory.
- CMLIST-CONTEXT - List the reopen copntext of a file.
- CMLIST-FILE - List cataloged files.
- CMLIST-HOST - List host file names.
- CMLIST-TAPE - List archived tape information.
- CMOFF - Terminate a catalog manager session.
- CMON - Initiate a catalog manager session.
- CMRENAME - Rename a cataloged file.
- CMSEARCH-ATTR - Retrieve archived files.
- CMSEARCH-NAME - Search the catalog by attribute.
- CMSEARCH-NAT - Search the catalog by name and attribute.
- CMSET - Set the default directory.
- CMSHOW - Show the default directory.
- CMTAPE - Clear tape information.
- CMTAPEDEL - Delete a tape from the catalog.
- CMUNCAT - Remove a name from the catalog.
- CMUNDEL - Undelete tape file(s) that are marked for deletion.

FORTRAN/C CALLABLE ROUTINES

- The same capabilities are provided to programmers by C and FORTRAN callable routines. These routines are used in LAS software to:
  - Catalog created host-files.
  - To remove deleted host-files from a catalog.
  - To obtain the associated host-file name for a TAE name.
  - To check file attributes before processing a file.
  - To archive files by CM conventions.
  - and generally to access the catalog to update or obtain file information.

FORTRAN Callable CM Functions

1	XEAOUT	- Output aliases and TAE names.
2	XEARCH	- Archive files.
3	XASOC	- Add and associated File to an attribute list.
4	XECAL	- Catalog and alias.
5	XECAT	- Catalog a file.
6	XECMIN	- Initialize user with catalog manager.
7	XEDEAS	- Delete associated files from an attribute list.
8	XEDEL	- Delete and uncatalog a file or branch.
9	XEDEL1	- Delete and uncatalog a single file.
10	XEDEUN	- Delete or uncatalog a single file.
11	XEDFLT	- Set default values.
12	XEHST	- Delete a host file.
13	XEEND	- Terminate communications with catalog manager.
14	XEERR	- Output a TAE Catalog Manager error message.
15	XEFID	- Get file ID of a cataloged file.
16	XEFILE	- Generate a host file name .
17	XEFULL	- Retrieve a fully qualified TAE name.
18	XEGET	- Retrieve the value of the specified attribute.
19	XEGTAF	- Retrieve associated file list.
20	XEGOUT	- Output alias and TAE names.
21	XEGPID	- Get process ID.
22	XEGTNM	- Get user name.
23	XEHOST	- Retrieve host file name of a cataloged file.
24	XEINIT	- Initialize user with catalog manager'.
25	XEIOP	- Open an image file for input.
26	XEIPOU	- Open an image file for output.
27	XELALS	- List aliases.
28	XELDIR	- Search by directory.
29	XELFIL	- Search by a file or branch name.
20	XELIAS	- Retrieve aliases of a TAE name.
21	XEMATR	- Modify the attributes of a cataloged file.
22	XEOUT	- Output a set of names.
23	XEPRNT	- Print name and/or attributes to terminal, file, or line printer.
24	XERALS	- Rename an alias.
25	XEROOT	- Set default root.
26	XERUIC	- Save user's indentification code.
27	XERETR	- Retrieve archived files.
28	XESATB	- Search by name and attributes.
29	XESATN	- Search by directory name.
30	XESET	- Set attributes of a file.
31	XESGSR	- Search by name and/or attributes; return a single name.
32	XESNAM	- Search by name only.
33	XESRCH	- Search by a file or branch name and/or arrtibutes.
34	XESTAR	- Initialize a stand-alone non-TAE user with the Catalog Manager.
35	XETNAM	- Get TAE name of an alias .
36	XEUNCAT	- Uncatalog a file.
37	XEUSER	- Get user information.
38	XEVFY	- Verify that a TAE data set exists in the catalog.

FILE ATTRIBUTES

- o Certain attributes apply to image data explicitly.
- o Off-line File Information: Tape Volume, Tape file name, ON/OFF line flags are now being maintained.
- o Four User defined attributes exist in the catalog.
  - LAS uses the DATA TYPE2 attribute to store "KEYS". These are used to access image label information from "keyed access" files in LAS.
- o CM allows file searching by name, attribute or both.

39 Catalog Manager Attributes

ATTRIBUTE DESCRIPTION	ATTRIBUTE	ATTRIBUTE TYPE	ACCESS
Data date	CMADAT	integer*2 (3)**	read-write
Data time	CMATIM	integer*2 (3)***	read-write
Center latitude	CMACLA	real*8	read-write
Delta latitude	CMADLA	real*8	read-write
Center longitude	CMACLO	real*8	read-write
Delta longitude	CMADLO	real*8	read-write
Data type 1	CMADT1	character*8	read-write
Data type 2	CMADT2	character*8	read-write
User field 1	CMAUS1	character*8	read-write
User field 2	CMAUS2	character*8	read-write
User field 3	CMAUS3	character*8	read-write
User field 4	CMAUS4	character*8	read-write
Version	CMAVER	character*1	read-write
File (F) or directory(C)	CMAFIL	character*1	read-write
Disk volume & serial #	CMAVID	character*12	read-write
Source code	CMASRC	integer	read-write
Instrument code	CMAINS	integer	read-write
Starting line number	CMASLN	integer	read-write
Number of lines	CMANLN	integer	read-write
Line increment	CMALNI	integer	read-write
Starting sample number	CMASSM	integer	read-write
Number of samples	CMANSM	integer	read-write
Sample increment	CMASMI	integer	read-write
Number of bands	CMANBD	integer	read-write
Spectral Organization	CMASPC	character*2	read-write
Project	CMAPRO	character*30	read-write
Host file creation date	CMACRD	integer*2 (3)**	read-write
Host file creation time	CMACRT	integer*2 (3)***	read-write
Last access date	CMALAD	integer*2 (3)**	read-write
Last access time	CMALAT	integer*2 (3)***	read-write
Modification date	CMAMOD	integer*2 (3)**	no access
Modification time	CMAMOT	integer*2 (3)***	no access
Footage	CMAFTG	integer	read only
Tape label	CMATLB	character*6	read only
Name of file on tape	CMATFL	character*19	read only
On-line indicator	CMAONL	logical	read only
On-tape indicator	CMAOFF	logical	read only
Marked for deletion flag	CMAMKD	logical	read only
Host filename w/o device	CMAHST	character*80	no access

** Dates are input as three words (two bytes each). The first word contains the year, the second contains the month, and the third contains the day of the month.

*** Times are input as three words (two bytes each). The first word contains the hour, the second contains the minute, and the third contains the seconds.

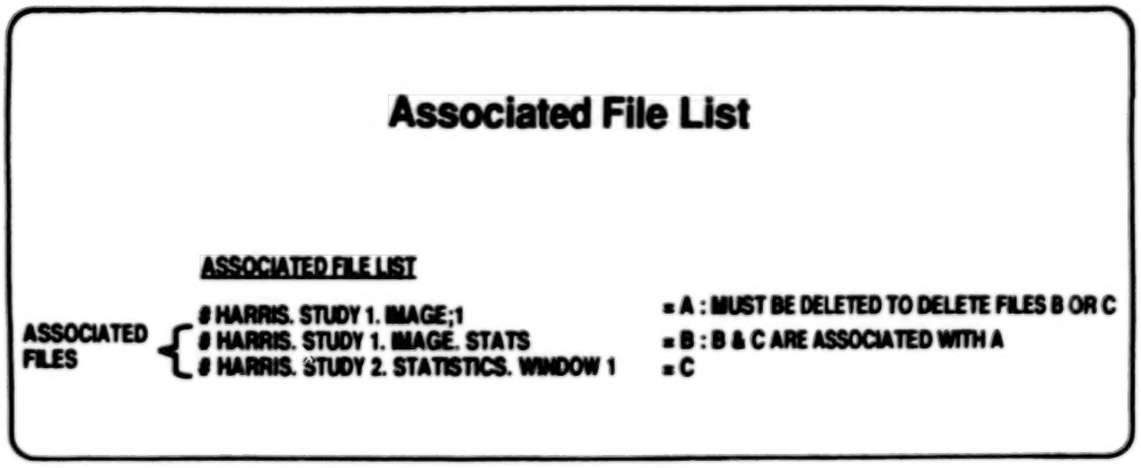
Associated File Lists

o Lists of associated files are kept with file attributes

  - Each File or directory may have up to 30 other file names kept in its associated file list.

  - An associated file can not be deleted untill the file it is associated with is deleted.

  - An example of the use of associated files in LAS will be shown.


Associated File List
ASSOCIATED FILE LIST
ASSOCIATED FILES
HARRIS. STUDY 1. IMAGE;1
HARRIS. STUDY 1. IMAGE. STATS
HARRIS. STUDY 2. STATISTICS. WINDOW 1
= A : MUST BE DELETED TO DELETE FILES B OR C
= B : B & C ARE ASSOCIATED WITH A
= C

ARCHIVE / RETRIEVE CAPABILITIES

o A user may archive or retrieve any cataloged file to or from tape. After the file has been archived it remains cataloged along with it's location on tape.

o The following actions are taken by CMARCH.

  - Searches User's Catalog for files meeting the name and attribute combination specified by the user.

  - Prompts the user to verify the file names it finds.

  - Requests operator to specify a tape drive and mount a tape Volume.

  - Checks that the correct tape volume is mounted.

  - Assigns tape Volume numbers to new tapes.

  - Requests operator for a new tape when requested or needed.

  - Requests the operator to mount a new tape if the end of tape if found while writing a file. This tape is handled as a "continuation" tape.

  - Writes the file or files from disk to tape.

  - Updates file attributes (ON/OFF line, tape volume, tape file name, etc.)

  - Deletes the on-line files if requested.

  - CMRETR performs an equivalent scenario for retrieving files from tape.

Archive and Retrieve Command Lines

```
TAE> CMARCH NAME      = TAE name +
            DELETE    = KEEP or DELETE
            VERIFY    = verify or no verify +
            ERROR     = continue or stop +
            TAPE      = unload or nounload +
            SUPERSED  = super or nosuper +
            NEWTAPE   = new or nonew +
            DATE      = date-string +
            PROJECT   = a string +
            INSTR     = a string +
            SOURCE    = a string +
            USER1     = a string +
            USER2     = a string +
            USER3     = a string +
            USER4     = a string +
            CRDATE    = createion-date +
            LADATE    = last-used-date
```

```
TAE> CMRETR NAME      = TAE name +
            DELETE    = KEEP or DELETE +
            VERIFY    = verify or no verify +
            ERROR     = continue or stop +
            TAPE      = unload or nounload +
            DATE      = date-string +
            PROJECT   = a string +
            INSTR     = a string +
            SOURCE    = a string +
            USER1     = a string +
            USER2     = a string +
            USER3     = a string +
            USER4     = a string +
            TAPEVOL   = vol-id   +
            CRDATE    = createion-date +
            LADATE    = last-used-date
```

o System Level Functions also exist to handle archive tapes. These functions:

- Maintain a log of tape ownership
- Allow users to delete a tape from their list of owned volumes.
- Allow tape compression of all user's files which are marked for deletion.
- Maintain a list of tape volumes which are free for use.
- Provide an option for operators to assign tape label names.
- and provide recovery procedures for corrupted or over-written CM tapes.

o The FORTRAN callable routine is used in LAS to handle multi-band images.

- Each band is stored in a separate file in LAS.
- Label information must be archived for each band.
- LAS prompts the user to verify a multi-band image name only, rather than for each individual band and label file.
- Online file deletion is postponed untill the entire multi-band image and associated label files are successfully Archived.

Summary of Catalog Manager System Functions

- Edit the catalog file (CMCATEDT)
- Verify the integrity of the catalog file (CMCATVFY)
- Copy a user's tapes onto a new set of tapes (CMCOMPRESS)
- Do free tape list functions (CMDOFREE)
- Delete the Catalog Manager process (CMDOWN)
- Generate a catalog file (CMGENCAT)
- List all the tapes in the Catalog (CMTAPELIST)
- Lock the catalog file for editing (CMLOCK)
- Log on as another user (CMONAS)
- Toggle Catalog Manager debug switchword (CMTOGGLE)
- Unlock the catalog file after editing (CMUNLOCK)
- Create the Catalog Manager process (CMUP)
- Add or delete a user from the catalog (CMUSER)
- Log off Catalog Manager users (REMUSR)
- Profile a Catalog Manager tape (PROFILE)
- Dump a Catalog Manager tape file onto disk in VMS format (RESTOREF)

LAS USER SCENARIO

o Tape ingest, subset, classification, registration, film output and archive.

  - Concentration on the use of directory names and file attributes for searching by example user operations.

LAS USER SCENARIO

DIRECTORY: #IRANI.PROJECTS.CLASS.

INGEST FROM TAPE	SUBSET STUDY AREA (copy window)
FULLTM.B1.P;1	WINDOW.B1;1
FULLTM.B2.P;1	WINDOW.B2;1
FULLTM.B3.P;1	WINDOW.B3;1
FULLTM.B4.P;1	WINDOW.B4;1
FULLTM.B5.P;1	WINDOW.B5;1
FULLTM.B7.P;1	WINDOW.B7;1

SCALE I*2 DATA RANGE TO BYTE (0-255)	CONVERT DATA TYPE TO BYTE
AREA.GDF;1	IMAGE.GDF;1
AREA.GM1;1	IMAGE.GM1;1
AREA.GM2;1	IMAGE.GM2;1
AREA.GM3;1	IMAGE.GM3;1
AREA.GM4;1	IMAGE.GM4;1
AREA.GM5;1	IMAGE.GM5;1
AREA.GM6;1	IMAGE.GM6;1

ANALYSE IMAGE	CREATE CLASSIED IMAGE
STATS1;1	LAND.K;1
STATS2;1	LAND.B;1

REGISTER TO MAP PROJECTION	CREATE DISPLAY IMAGE
MAP.TIEPTS;1	COLOR.LUK
LUK.TIEPTS;1	COLOR.LUB
LUK.GRID	LUK.LUT
LUK.REG	LUB.LUT
LUB.REG	

```
CMLIST-FILE NAME=*

$IRANI.BENCH1.GM1;01
$IRANI.BENCH1.GM2;01
$IRANI.BENCH1.GM3;01
$IRANI.BENCH1.GM4;01
$IRANI.BENCH1.GM5;01
$IRANI.BENCH1.GM6;01
$IRANI.BENCH1;01
$IRANI.CAL.GRP;01
$IRANI.CAL.IMG;01
$IRANI.MTF.GRP;01
$IRANI.MTF;01
$IRANI.PROJECTS.CLASS.AREA.GM1;01
$IRANI.PROJECTS.CLASS.AREA.GM2;01
$IRANI.PROJECTS.CLASS.AREA.GM3;01
$IRANI.PROJECTS.CLASS.AREA.GM4;01
$IRANI.PROJECTS.CLASS.AREA.GM5;01
$IRANI.PROJECTS.CLASS.AREA.GM6;01
$IRANI.PROJECTS.CLASS.AREA;01
$
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
$IRANI.PROJECTS.CLASS.AREA.GM6;01
$IRANI.PROJECTS.CLASS.AREA;01
$IRANI.PROJECTS.CLASS.COLOR.LUB;01
$IRANI.PROJECTS.CLASS.COLOR.LUK;01
$IRANI.PROJECTS.CLASS.FULLTM.B1.P;01
$IRANI.PROJECTS.CLASS.FULLTM.B2.P;01
$IRANI.PROJECTS.CLASS.FULLTM.B3.P;01
$IRANI.PROJECTS.CLASS.FULLTM.B4.P;01
$IRANI.PROJECTS.CLASS.FULLTM.B5.P;01
$IRANI.PROJECTS.CLASS.FULLTM.B7.P;01
$IRANI.PROJECTS.CLASS.IMAGE.GM1;01
$IRANI.PROJECTS.CLASS.IMAGE.GM2;01
$IRANI.PROJECTS.CLASS.IMAGE.GM3;01
$IRANI.PROJECTS.CLASS.IMAGE.GM4;01
$IRANI.PROJECTS.CLASS.IMAGE.GM5;01
$IRANI.PROJECTS.CLASS.IMAGE.GM6;01
$IRANI.PROJECTS.CLASS.IMAGE;01
$IRANI.PROJECTS.CLASS.LAND.B;01
$IRANI.PROJECTS.CLASS.LAND.K;01
$IRANI.PROJECTS.CLASS.LUB.LUT;01
$IRANI.PROJECTS.CLASS.LUB.REG;01
$IRANI.PROJECTS.CLASS.LUK.GRID;01
$IRANI.PROJECTS.CLASS.LUK.LUT;01
$IRANI.PROJECTS.CLASS.LUK.REG;01
```

```
@IRANI.PROJECTS.CLASS.LUK.TIEPTS;01
@IRANI.PROJECTS.CLASS.MAP.TIEPTS;01
@IRANI.PROJECTS.CLASS.STATS1;01
@IRANI.PROJECTS.CLASS.STATS2;01
@IRANI.PROJECTS.CLASS.WINDOW.B1;01
@IRANI.PROJECTS.CLASS.WINDOW.B2;01
@IRANI.PROJECTS.CLASS.WINDOW.B3;01
@IRANI.PROJECTS.CLASS.WINDOW.B4;01
@IRANI.PROJECTS.CLASS.WINDOW.B5;01
@IRANI.PROJECTS.CLASS.WINDOW.B7;01
@IRANI.ROSE.V01.LAB;01
@IRANI.ROSE;01
@IRANI.STATS1;01
@IRANI.TIEPTS2;01
@IRANI.UNIFORM.GRID;01
@IRANI.UNIFORM.GRP;01
@IRANI.WASH.CLASS;01
@IRANI.WASH.CLASS;02
@IRANI.WASH.QUARTER1.B1.P;01
@IRANI.WASH.QUARTER1.B2.P;01
@IRANI.WASH.QUARTER1.B3.P;01
@IRANI.WASH.QUARTER1.B4.P;01
@IRANI.WASH.QUARTER1.B5.P;01
@IRANI.WASH.QUARTER1.B6.P;01
```

```
TAE>CMSET @IRANI.PROJECTS

)CMSEARCH-ATTR NAME=* DATA1=GDF

@IRANI.PROJECTS.CLASS.AREA;01
@IRANI.PROJECTS.CLASS.IMAGE;01



)CMSEARCH-ATTR NAME=@IRANI.* DATA1=GDF

@IRANI.BENCH1;01
@IRANI.CAL.GRP;01
@IRANI.MTF.GRP;01
@IRANI.PROJECTS.CLASS.AREA;01
@IRANI.PROJECTS.CLASS.IMAGE;01
@IRANI.UNIFORM.GRP;01

)
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
)CMLIST-DIR DIR=*

$IRANI.PROJECTS.CLASS.AREA
$IRANI.PROJECTS.CLASS.COLOR
$IRANI.PROJECTS.CLASS.FULLTM
$IRANI.PROJECTS.CLASS.IMAGE
$IRANI.PROJECTS.CLASS.LAND
$IRANI.PROJECTS.CLASS.LUB
$IRANI.PROJECTS.CLASS.LUK
$IRANI.PROJECTS.CLASS.MAP
$IRANI.PROJECTS.CLASS.STATS1
$IRANI.PROJECTS.CLASS.STATS2
$IRANI.PROJECTS.CLASS.WINDOW


)CMLIST-DIR DIR=*.LUK.*

$IRANI.PROJECTS.CLASS.LUK.GRID;01
$IRANI.PROJECTS.CLASS.LUK.LUT;01
$IRANI.PROJECTS.CLASS.LUK.REG;01
$IRANI.PROJECTS.CLASS.LUK.TIEPTS;01
)
```

```
)CMSEARCH-ATTR NAME=$CROSSCUT  DATA1=GDF

$CROSSCUT.BIG.GRP;01
$CROSSCUT.DROSE;01
$CROSSCUT.FALL;01
$CROSSCUT.MONKEY;01
$CROSSCUT.OHIO.GDF;01
$CROSSCUT.RAND.IMG;01
$CROSSCUT.SANFALL.GDF;01
$CROSSCUT.SANFRAN.GDF;01
$CROSSCUT.SLICE.GDF;01
$CROSSCUT.SLICEZ.GDF;01
$CROSSCUT.WIDE;01
$CROSSCUT.X.SLICE;01
$CROSSCUT.XSLICE.GDF.KAR;01
$CROSSCUT.XSLICE.GDF.KAR;02
$CROSSCUT.XSLICE.GDF;01
$CROSSCUT.XSLICE7.GDF;01
$CROSSCUT.X1000.KAR.KART;01
$CROSSCUT.X1000.KAR;01
$CROSSCUT.X1000.KAR;02
$CROSSCUT.X1000;01
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
[redacted] proc "CMCHGATR", library "CMUSREXE"                    Pg 1+

Change Attributes of a Cataloged File

 parm      description                          value

NAME       Data File or Directory Name
             (No wildcards allowed)


EDIT       Edit Attributes                      "EDIT"
           EDIT or NOEDIT

DATE       Data Date (ex. 14-NOV-1984)          --  (null value)

TIME       Data Time (ex.  06:06:38)            --  (null value)

CLAT       Center Latitude (-90 to +90)         --  (null value)

Enter: parm=value,HELP,PAGE,QUALIFY,SHOW,RUN,EXIT,SAVE,RESTORE; RETURN to page.
?
```

```
[redacted] proc "CMCHGATR", library "CMUSREXE"                    Pg 2+

Change Attributes of a Cataloged File

 parm      description                          value

DLAT       Delta Latitude                       --  (null value)

CLON       Center Longitude (-180 to +180)      --  (null value)

DLON       Delta Longitude                      --  (null value)

STARTLIN   Starting line number                 --  (null value)

LINES      Number of lines                      --  (null value)

LINEINC    Line increment                       --  (null value)

STARTSMP   Starting sample number               --  (null value)

Enter: parm=value,HELP,PAGE,QUALIFY,SHOW,RUN,EXIT,SAVE,RESTORE; RETURN to page.
?
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
[redacted] proc "CMCHGATR", library "CMMUSREXE"                    Pg 3+

Change Attributes of a Cataloged File

 parm      description                          value

SAMPLES    Number of samples per line      -   (null value)

SAMPINC    Sample increment                -   (null value)

BANDS      Number of bands                 -   (null value)

SPECORG    Spectral organization           -   (null value)

PROJECT    Project name -                  -   (null value)
           (1 to 30 chars.)

INSTR      Instrument number (-128 to 127) -   (null value)


Enter: parm=value,HELP,PAGE,QUALIFY,SHOW,RUN,EXIT,SAVE,RESTORE; RETURN to page.
?
```

```
[redacted] proc "CMCHGATR", library "CMMUSREXE"                    Pg 4+

Change Attributes of a Cataloged File

 parm      description                          value

SOURCE     Source Number (-128 to 127)     -   (null value)

DATA1      Data Type 1 (1 to 8 chars.)     -   (null value)

DATA2      Data Type 2 (1 to 8 chars.)     -   (null value)

USER1      User Field 1 (1 to 8 chars.)    -   (null value)

USER2      User Field 2 (1 to 8 chars.)    -   (null value)

USER3      User Field 3 (1 to 8 chars.)    -   (null value)

USER4      User Field 4 (1 to 8 chars.)    -   (null value)

Enter: parm=value,HELP,PAGE,QUALIFY,SHOW,RUN,EXIT,SAVE,RESTORE; RETURN to page.
?
```

```
proc "CMCHGATR", library "CMUSREXE"                    Pg 5+

Change Attributes of a Cataloged File

 parm      description                        value

VERSION   BEST for Best Version          —   (null value)

Enter: parm=value,HELP,PAGE,QUALIFY,SHOW,RUN,EXIT,SAVE,RESTORE; RETURN to page.
```

```
proc "CMCHGATR", library "CMUSREXE"                    Pg 6.

Change Attributes of a Cataloged File

 parm      description                        value

ASSOC     Associated File(s)               "@IRAMI.PROJECTS.CLASS.LUK (1)
          (maximum of 30 entries;          .REG"
          valid for files only)            "@IRAMI.PROJECTS.CLASS.LUK (2)
                                           .LUT"
                                           "@IRAMI.PROJECTS.CLASS.LUK (3)
                                           .TIEPTS"
                                           "@IRAMI.PROJECTS.CLASS.LUK (4)
                                           .GRID"
                                           "@IRAMI.PROJECTS.CLASS.COL (5)
                                           OR.LUK"
                                           [redacted]          (6)
                                                               (7)
                                                               (8)+

[TAE-TNODEF] No default value defined for 'ASSOC(6)'.
Enter: parm=value,HELP,PAGE,QUALIFY,SHOW,RUN,EXIT,SAVE,RESTORE; RETURN to page.
ASSOC(6)=@IRAMI.PROJECTS.CLASS...
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
[redacted] proc "CHCHGATR", library "CMMUSREXE"                    Pg 6.

Change Attributes of a Cataloged File

 parm        description                          value

ASSOC    Associated File(s)                  "$IRANI.PROJECTS.CLASS.LUK (1)
         (maximum of 30 entries;             .REG"
         valid for files only)               "$IRANI.PROJECTS.CLASS.LUK (2)
                                             .LUT"
                                             "$IRANI.PROJECTS.CLASS.LUK (3)
                                             .TIEPTS"
                                             "$IRANI.PROJECTS.CLASS.LUK (4)
                                             .GRID"
                                             "$IRANI.PROJECTS.CLASS.COL (5)
                                             OR.LUK"
                                                                    (6)
                                                                    (7)
                                                                    (8)+

Enter: parm=value,HELP,PAGE,QUALIFY,SHOW,RUN,EXIT,SAVE,RESTORE; RETURN to page.
[redacted]SAVE ARCSET
```

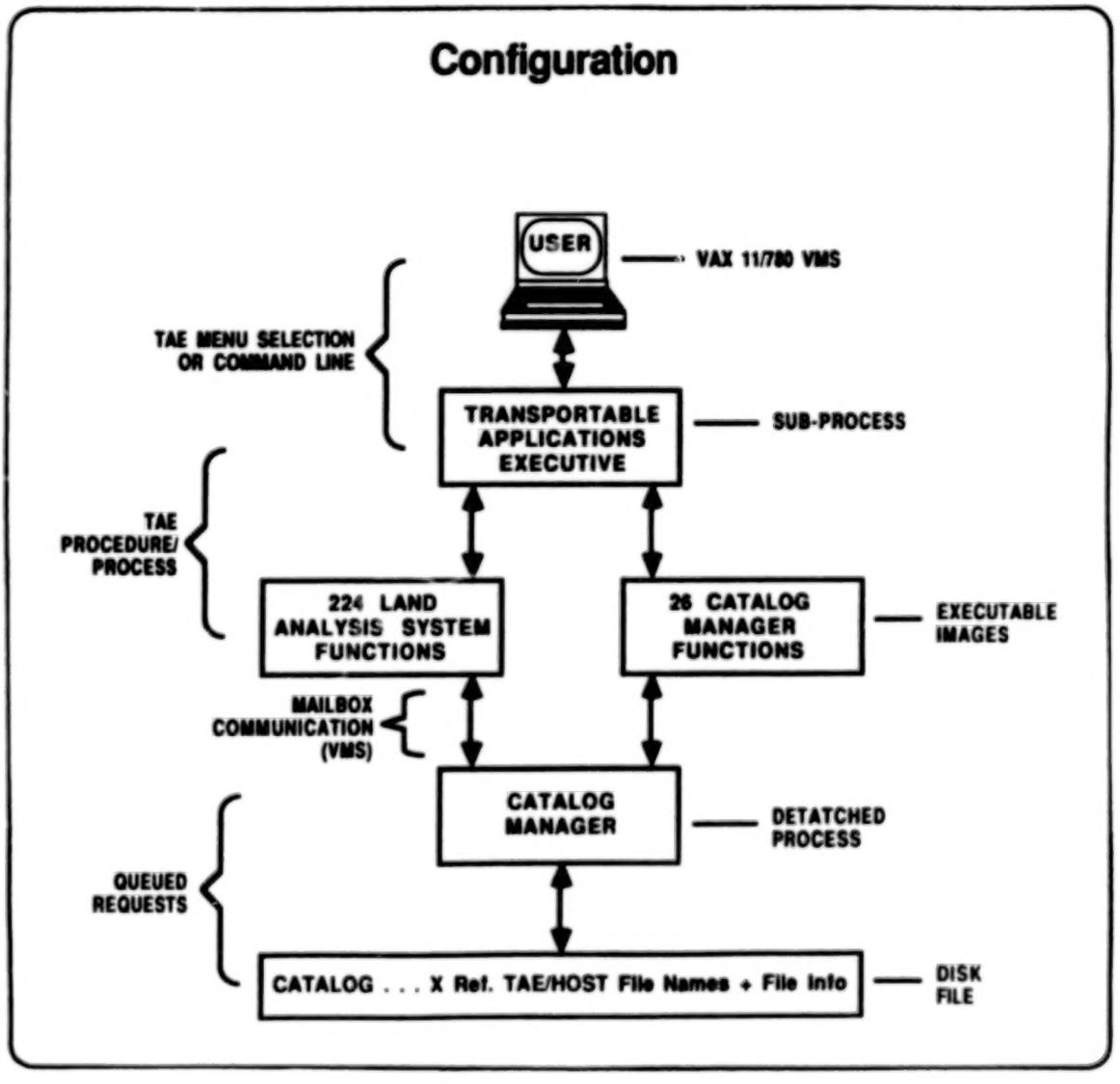
```
[redacted] proc "CHCHGATR", library "CMMUSREXE"                    Pg 6.

Change Attributes of a Cataloged File

 parm        description                          value

ASSOC    Associated File(s)                  "PROJECTS.CLASS.STATS1"    (1)
         (maximum of 30 entries;             "PROJECTS.CLASS.LAND.K"    (2)
         valid for files only)               "PROJECTS.CLASS.LUK.GRID"  (3)
                                             "PROJECTS.CLASS.COLOR.LUK" (4)
                                             "PROJECTS.CLASS.LUK.LUT"   (5)
                                                                        (6)
                                                                        (7)
                                                                        (8)
                                                                        (9)
                                                                        (10)
                                                                        (11)
                                                                        (12)
                                                                        (13)+

Enter: parm=value,HELP,PAGE,QUALIFY,SHOW,RUN,EXIT,SAVE,RESTORE; RETURN to page.
[redacted]RESTORE LANDSET
```

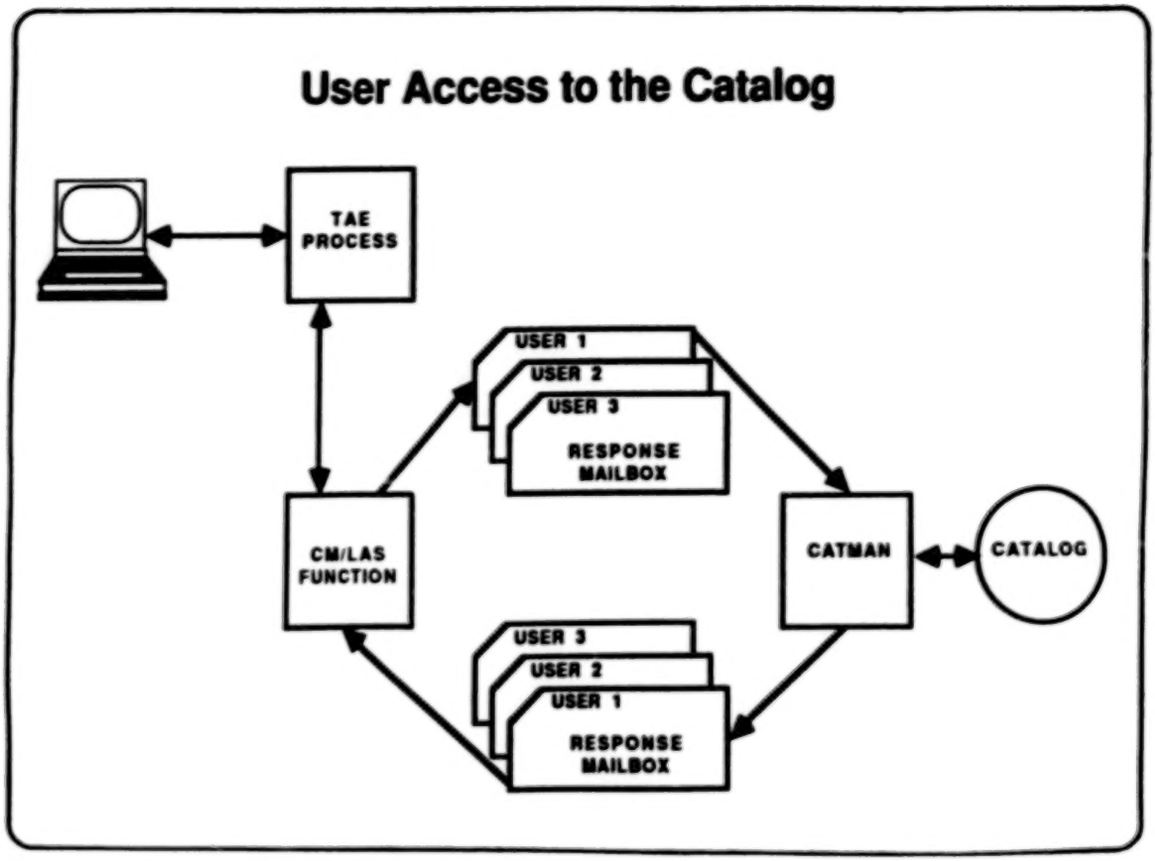
```
$IRANI.PROJECTS.CLASS.IMAGE;01

CM Name Type      = file              Disk Volume ID    = LAUSR1
Tape Label        =                   Tape copy status  = nonexistent
Creation Date     =  2-OCT-1986       Creation Time     = 17:34:41
Last Access Date  =  3-OCT-1986       Last Access Time  = 11:30:59
Date              =                   Time              =
Center Latitude   =                   Delta Latitude    =
Center Longitude  =                   Delta Longitude   =
Starting Line     =                   Number of Lines   =
Line Increment    =                   Starting Sample   =
Number of Samples =                   Sample Increment  =
Number of Bands   =                   Spectral Org.     =
Source            =                   Instrument        =
Data Type 1       = GDF               Data Type 2       = 34R0A6A2
User Field 1      =                   User Field 2      =
User Field 3      =                   User Field 4      =
Version           =
Project           =

Enter E to exit or press RETURN to continue:
```

```
Center Longitude  =                   Delta Longitude   =
Starting Line     =                   Number of Lines   =
Line Increment    =                   Starting Sample   =
Number of Samples =                   Sample Increment  =
Number of Bands   =                   Spectral Org.     =
Source            =                   Instrument        =
Data Type 1       = GDF               Data Type 2       = 34R0A6A2
User Field 1      =                   User Field 2      =
User Field 3      =                   User Field 4      =
Version           =
Project           =

Enter E to exit or press RETURN to continue:

Associated Files:

$IRANI.PROJECTS.CLASS.STATS1
$IRANI.PROJECTS.CLASS.LAND.K
$IRANI.PROJECTS.CLASS.LUK.GRID
$IRANI.PROJECTS.CLASS.COLOR.LUK
$IRANI.PROJECTS.CLASS.LUK.LUT

Enter E to exit or press RETURN to continue:
```

CONFIGURATION

- o Hardware.
- o TAE Sub-process.
- o LAS and CM executable images
- o Communication via system-service mailbox under VMS.
- o Catalog on disk
- o One user request at a time.





PERFORMANCE

- The Catalog can be corrupted via hardware problems such as a head-crash.
- System functions exist to verify catalog integrity on a regular basis.
- System Functions Exist to Edit the Catalog to recover from corruption.
- It has been suggested that the catalog could be decentralized, e.g., maintain one catalog per user. This would lessen impact of catalog corruption on all users and the system as a whole. Right now, if CM Goes Down, so does LAS. There may be a problem, however, in accessing files across user roots in a decentralized catalog.
- Mailbox Communication may slow down the performance of LAS by queueing one user for catalog access at a time. This occurrs during times of heavy system load at Goddard. Decentralizing the catalog may reduce the mailbox communication bottle-neck.
- EDC is currently working on an "all new" version of CM using a fresh design concept.

N87
23164
UNCLAS

N87-23164

DISPLAY MANAGEMENT SUBSYSTEM, VERSION 1
A USER'S EYE VIEW

Dolores Parker
Image Analysis Facility
Space Data and Computing Division
NASA/Goddard Space Flight Center

Display Management Subsystem

NASA

A User's Eye View

- A Case for DMS
- Design and Functionality of DMS
- Support for DMS
- System Information
- Future Directions
- Summary

Display Management Subsystem

NASA

Typical Image Processing Environment

User

Image Data

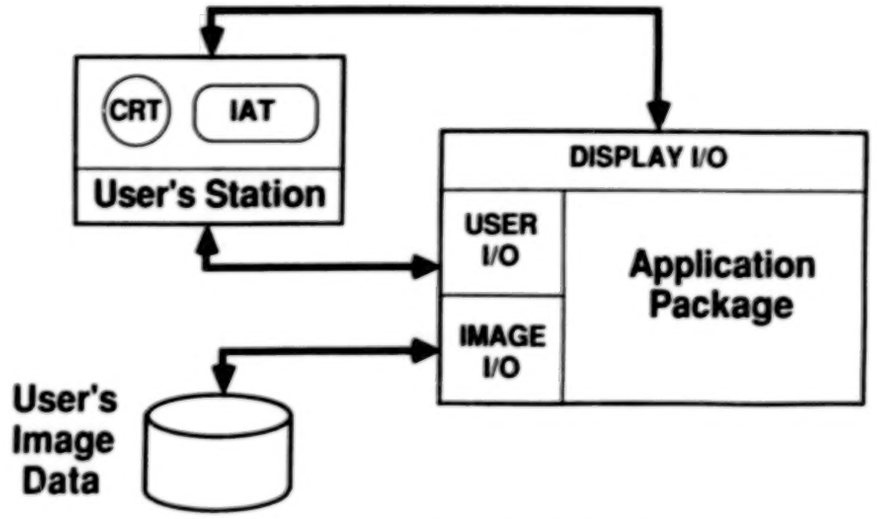
Image Analysis Terminal

Image Processing Software

Display
Management
Subsystem

NASA

Typical Image Processing Environment

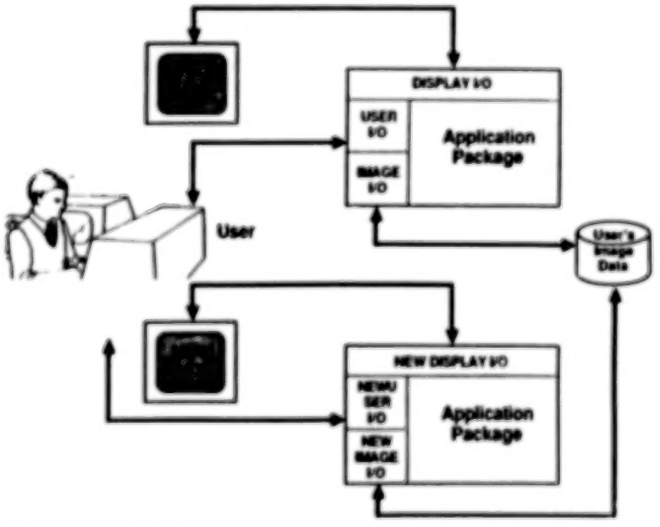


Display
Management
Subsystem

NASA

Typical Image Processing Environment

System Upgrade: A New IAT Is Added



Display
Management
Subsystem

NASA

Without DMS

Display
Management
Subsystem

NASA

Enter DMS

A Layered Software Design

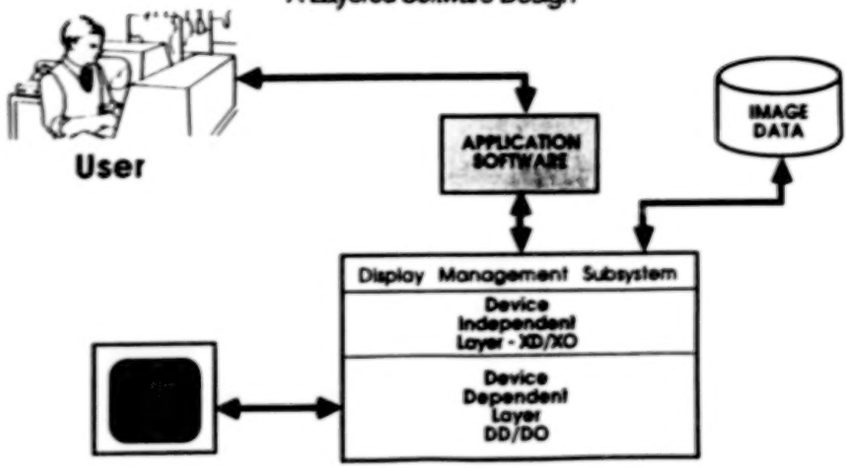
Display Management Subsystem
Device Independent Layer - XD/XO
Device Dependent Layer DD/DO

Display
Management
Subsystem

NASA

ENTER DMS

A Layered Software Design

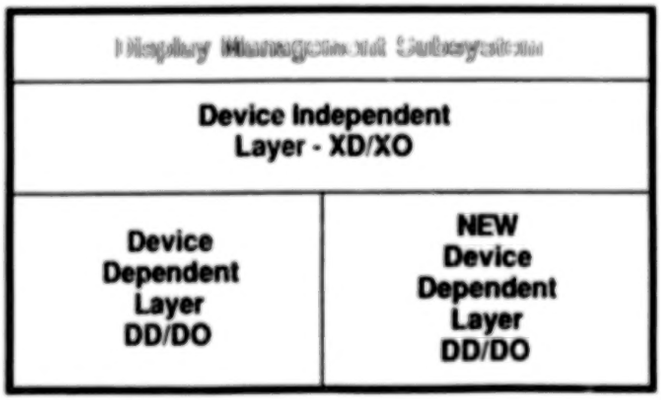


Display
Management
Subsystem

NASA

Enter DMS

System Upgrade: A New IAT Is Added

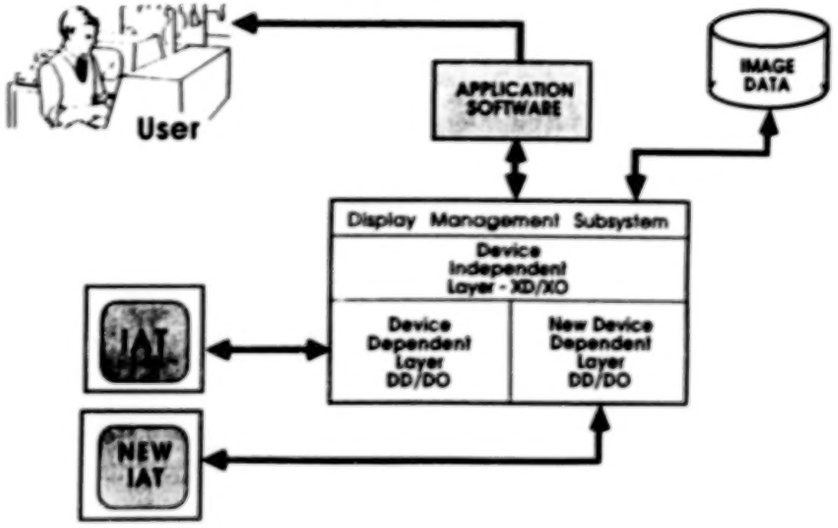

Display Management Subsystem
Device Independent Layer - XD/XO
Device Dependent Layer DD/DO
NEW Device Dependent Layer DD/DO

Display
Management
Subsystem

NASA

ENTER DMS

System Upgrade: A New IAT Added



Display
Management
Subsystem

NASA

With DMS

Display Management Subsystem

NASA

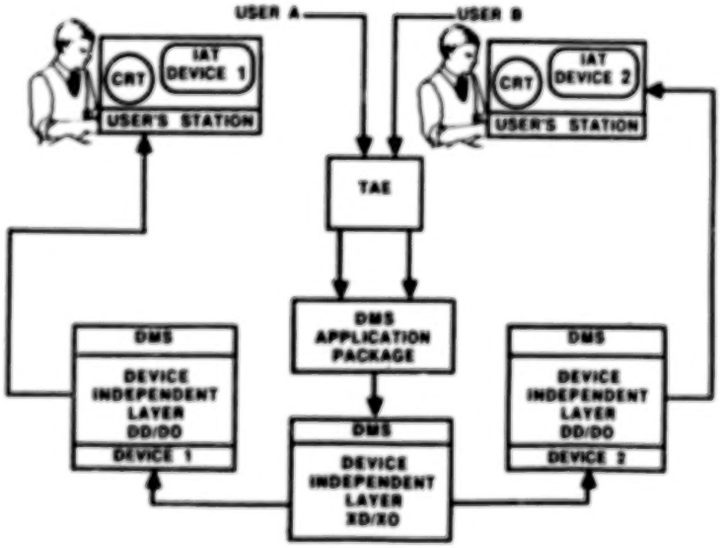
What Is DMS?

- Subsystem of TAE
- Device-Independent Interface
- Layered Software Design
- A Solution

Display Management Subsystem

NASA

• Subsystem of TAE



Display
Management
Subsystem

NASA

• Device Independent Interface

Provides Generic Services

- Initiation and Termination
- Image Transfer and Setup
- Image Viewing/Alteration
- Image Manipulation
- Overlay Plane Support
- Cursor/Interrupt Support

Display
Management
Subsystem

NASA

• Layered Software Design

Upper Layer

- Device Independent
- Generic Services
- C and FORTRAN Callable

Lower Layer

- Device Dependent
- Specific Services
- Data Structure Management
- Image I/O Support

Display
Management
Subsystem

NASA

• A Solution

- Simple to Use
- Generic Applications Available for Performing Many Image Processing Tasks
- Easily Expandable to Meet Local Needs
- Callable by C and FORTRAN Applications
- Portable to Accommodate Different IAT's
- Modeling of Device-Specific and System-Specific Elements

Display
Management
Subsystem

NASA

• Structure of DMS

Device Independent Layer

CXD/CXO	FXD/FXO	GENERIC SERVICES
C INTERFACE	FORTRAN INTERFACE	

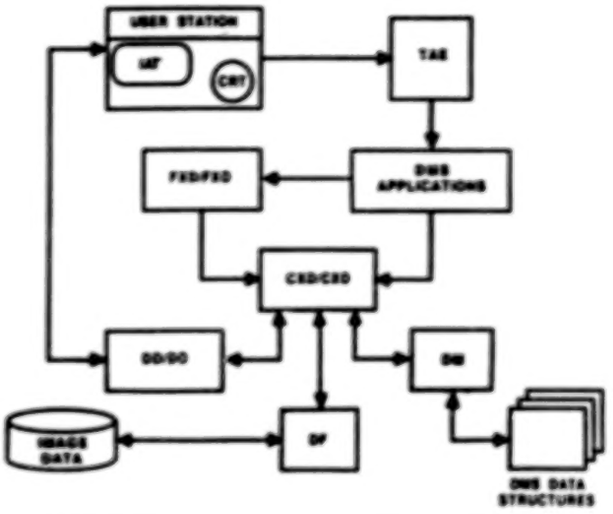
Device Dependent Layer

DD/DO	DM	SPECIFIC SERVICES
DEVICE INTERFACE	DATA STRUCTURE MANAGEMENT	

Display
Management
Subsystem

NASA

Structure of DMS



Display
Management
Subsystem

NASA

Application Functions

- 32 Currently Available
- Developed at EROS Data Center
- Coming Attraction: Mensuration Package

Display
Management
Subsystem

NASA

Application Functions

Some Examples

ALLOC	- Allocate a Display Device
CURSOR	- Turn the Cursor On or Off
TODSP	- Transfer a Disk Image File to Display Memory
SAVIMG	- Create an Entry for the Viewed Image in the ConfigurationTable
FLICKR	- Flicker Several Images on the Display
HISTO	- Draw a Histogram of an Image
PIVOT	- Pivot an Image
SHOIMG	- View an Image in Display Memory
ZOOPAN	- Expand or Pan a Portion of the Viewed Image Using a Pointing Device

Display
Management
Subsystem

NASA

DMS User Support

- Beta Test Sites With IIS
- Documentation
- GSFC User Support Office
  Phone (301) 286-6034
- Report Problems to USO
- User Network
- Request DMS thru USO
- Development is Ongoing

Display
Management
Subsystem

NASA

Details About the System

- Operates Under VMS and UNIX
- Uses 36,000 Disk Blocks
- Supports 3 Display Devices
  - IIS
  - RASTERTEK
  - DEANZA

Display
Management
Subsystem

NASA

Future Directions

- Generic Image I/O
- Multiple Device Handling
- Broader Graphics Support
- Single-User DMS

Display
Management
Subsystem

NASA

Summary

Why Does a User Want DMS?

- **Ease of Use**
- **Stability**
- **Many Application Functions Available**

Display
Management
Subsystem

NASA

Summary

Why Does an Application Programmer Want DMS?

- **Ease of Application Development**
- **Flexibility**
- **Interfaces With C and FORTRAN**

Display
Management
Subsystem

NASA

Summary

Why Does a System Developer Want DMS?

- **Ease of Expansion**
- **Models System-Dependent Elements**
- **System Support Available**
- **Longevity/Durability**

IMAGING APPLICATIONS OF TAE

N87
23165
UNCLAS

N87-23165

RATIONALIZING VICAR WITHIN A TAE FRAMEWORK–SOME PROBLEMS AND SOME SOLUTIONS

Colin P. Harris
Atmospheric Physics Group
Imperial College of Science and Technology
London SW7 2BZ

PRECEDING PAGE BLANK NOT FILMED

Abstract.

TAE implementation may impose a strain on centres with modest resources. This may be eased in a number of ways. The balance of a small number of expert users and a large number of computing novices at IPIPS imposes special constraints. Some solutions to these and other particular problems are described.

1. Introduction.

In many environments where complex applications software is used, implementation of a new system may inconvenience users and systems staff alike. Where there is no shortage of staff or time, a TAE-based system may be fully constructed and tested before being released to users. During this period, however, any contruction which reduces the strain of transition for both staff and users is clearly desirable. During the implementation of TAE on the IPIPS image processing system at Imperial College a number of such constructions were learnt. In addition, the selected TAE configuration had to encompass the particular needs of two user groups, and to complement the existing hardware and software.

2. The IPIPS system.

a. The target community.

As described by Grove (1985), the IPIPS system has two complementary user communities. Around one third of the users are researchers with varying amounts of experience who need access to the image processing subroutines in order to produce their own specialised software. The majority of the rest are masters degree students in remote sensing, around 25 in all, with little or no computing experience. Finally, there are visitors and external users, using the IPIPS system as a facility for their own work. People in this group need an intermediate level of user-friendliness. They may have quite complex needs, and may be very experienced on other systems, but will need guidance to find

the appropriate programs from the large libraries available.

The interests of the IPIPS users are wide-ranging. The original system was developed for terrestrial meteorology and Voyager image analysis, but subsequent involvement with the Centre for Remote Sensing at Imperial College has taken it into many more areas. These include Earth resources, ocean colour, pattern recognition and texture analysis, among others.

b. The hardware configuration.

The core of the IPIPS system (Hunt et al. 1985) is a DEC VAX 11/780 general purpose computer, with storage consisting of two 1600 bpi tape drives and four user accessable disks in addition to the system disk. Output devices include printer, plotter and video and photographic facilities. In addition there are two I2S dedicated image processing systems. These consist of user interface (trackball or tablet), colour monitor, and a processing unit coupled to a set of refresh memories, each of which may contain a 512x512x8 bit image. The Model 70E has 6 such channels whilst the newer Model 75 I2S has 15. Each I2S is capable of rapid and complex manipulation of single frames or sequences of images, in black and white or colour. The processing unit permits operations such as image convolution to be carried out in hardware, independently of the VAX. The I2S system is particularly suited to applications in Earth resources and meteorology. For a fuller description of the I2S capabilities see Adams and Driscoll (1979).

c. The software configuration.

The software available on the IPIPS system before the implementation of TAE was based around the VICAR system for image processing and parameterring. This was originally transported from the JPL IBM in the late 1970s (see Castleman 1979 for a description of the VICAR system). A transportable VAX version was subsequently developed by IPIPS (Lawden and Pearce 1980). This was complemented by a database system (G-EXEC), by CORE graphics, and by the online image manipulation facilities provided by the I2S. The greatest

single problem was the partial incompatibility of these systems, particularly in their different parameter retrieval routines.

3. Implementing TAE.

The target system to replace the existing software consisted of a division of the VICAR labour between TAE (handling parametering) and BISHOP (a superset of VICAR consisting of a selection of the image-processing subroutines and utilities) (Grove 1985). In addition the parametering routines associated with the other parts of the system were to be hidden beneath a TAE interface, enabling a common mode of user access to all parts of the system. In developing this system a major problem became apparent in that IPIPS simply did not have the manpower to fully convert large parts of the system to TAE in a short space of time. In the interim the less experienced users were having to cope with software wherein some programs were available only under the old arrangement and others only under the new. This is a problem that may potentially affect any system with limited resources.

To escape this problem a two stage implementation philosophy was used. The source of the Vicar programs was initially left untouched, but TAE was used on top of this to enable users to take advantage of the associated menus, tutor modes and second level help. This was done by generating a DCL VICAR command line within a TAE Procedure from concatenated parameter names and values. This command line was then used to run the program. Listing 1 provides an illustration of this.

This strategy proved extremely successful in smoothing the transition to TAE. The pressure on the systems staff to make the change as rapidly as possible was lifted as large numbers of programs could be made available to users in a relatively short period of time. The staff could then replace the temporary Procedures with Process PDFs and upgrade the Fortran source code from VICAR to BISHOP in their own time, and in a way that was invisible to the users. This process is still continuing.

4. a. Further enhancements.

In addition to this interim application, this method led to the implementation of many DCL commands (such as Link, Run etc.) within TAE. Again this was done through Procedures. This was found to have two advantages. Firstly, the user did not have to go through the sequence of commands DCL - <command> - TAE every time he wished to use a sequence of VMS commands. More importantly, only selected parameter qualifiers were made available in TAE. In the Link command, for instance, only /MAP, /DEBUG and /CROSS were used. This guided novice users who may have been lost within the extensive VMS help towards the parameters most likely to help them. See listing 2 for an example.

In addition to the general upheaval, the transition to TAE created some particular problems, and opportunities to remedy some old difficulties. The most ambitious development was a model-independent I2S driver. Applications programs link to a dummy shareable image at link time and to the appropriate (Model 70 or Model 75) real shareable image at run time. A TAE procedure selects the correct image by setting an appropriate global symbol that acts as a pointer. The shareable images associated with each I2S contain subroutines with the same name and parameter list, so that a program calling just these model-independent routines will work equally well on either I2S. The I2S primitive subroutines are different for the two models, and previously a user would have had to write a different program for each I2S. These differences are now hidden from the user. In principle a complete graphics system may be built in this way, with a set of shareable images corresponding to the output devices. This has a number of advantages. Firstly, there is a considerable saving in space since the executable images no longer contain the code to drive all the devices. Secondly, upgrades to individual devices' code may be made without rebuilding the whole system. Finally, new output devices may be added simply by adding new shareable images to the set, without disturbing the existing code. The CORE system might particularly lend itself to this architecture on the VAX, and IPIPS might consider this development at a later stage.

The implementation of TAE enabled simple menus to be constructed as another major development. Even the most experienced users were unfamiliar with all of the 150 plus applications programs available at IPIPS. Helping new and external users to find the programs they needed had long been a problem, and TAE menu trees arranged by application has made a great difference in this area.

In another area, TAE tutor mode has had an unexpectedly beneficial effect. VICAR programs may frequently have 10 or more optional parameters, and occasionally as many as 100. The well-known dislike of ploughing through many pages of hardcopy had always meant that users rarely used the more obscure parameters, although these were often very useful. Tutor mode has greatly increased user awareness of the system's capabilities, and this has in turn made for a more productive usage.

b. An outstanding problem.

The dual upgrade to Version 4.1 VMS and Version 1.3 TAE has induced one problem. Version 4.1 has been observed on many machines to run more slowly than earlier versions. This problem is made worse by the extra processes created by TAE. The observed additional overhead in response time for a given program varies between 5 % and as much as 25 %. The larger values are associated with programs involving user IO, whilst programs that involve mainly calculation are virtually unaffected.

5. Conclusions.

In summary, the TAE system is now essentially fully implemented as part of the IPIPS image processing system. This has been accomplished with only limited resources but the minimum of user disruption. The future development of such systems will surely hinge around the concept of integration. It is essential that the image handling, graphics and database parts of such complex systems are assembled and presented to the users in a coherent fashion. This is a path that systems such as IPIPS, MIPL at JPL, and the LAS all seem to be following. Future systems such as the Galileo HIIPS

system will also need this coherence, and it is apparent that the TAE Executive is currently the best way to provide it.

Refs.

Adams J.R. and Driscoll E.C., "A low cost transportable image processing system," 1st ASSP WKSHP ON 2-D DIG. SIG. PROC., 1979.

Castleman, K.R., "Digital image processing,"
Prentice-Hall, 1979

Grove, L, "TAE and BISHOP in a Teaching Environment," Fifth TAE Users Conference, Maryland (1985).

Hunt G.E., Barrey R.F.T., Clark D.R., Easterbrook M., Gorley R., Marriage N., Muller J-P.A.L., Roff C.E. and Rumball D., "The Interactive Planetary Image-Processing System," IEEE Transactions on Geoscience and Remote Sensing, Vol. GE-23, No 4, July 1985.

Lawden M.J. and Pearce D., "Making the VICAR system portable,"
J.B.I.S. 33, 369-376, 1980

LISTING 1 - AN IMAGE COPYING ROUTINE

```
PROCEDURE HELP=*
PARM (IN,OUT) TYPE=(STRING,80)
PARM SIZE TYPE=INTEGER COUNT=(0,4) DEFAULT=--
LOCAL COMMAND TYPE=(STRING,130) INITIAL="DCL ACOPY "
LOCAL SIZIN TYPE=(STRING,130)
LOCAL NEWSIZ TYPE=INTEGER
LOCAL I TYPE=INTEGER INITIAL=1
BODY
IF ( $COUNT(SIZE) <> 0 )
     LET SIZIN=" SIZE="
     LOOP
          IF ( I > $COUNT(SIZE) ) BREAK
          LET NEWSIZ=SIZE(I)
          LET SIZIN="&SIZIN" // "&NEWSIZ"
          IF ( I <> 4 )
              LET SIZIN="&SIZIN" // ","
          END-IF
          LET I=I+1
     END-LOOP
     LET COMMAND="&COMMAND" // "&SIZIN"
END-IF
DCL ACOPY:==$P$:[VICLIB.PROGRAMS]ACOPY.EXE
&COMMAND IN="&IN" OUT="&OUT"
END-PROC
.TITLE
A PROC  to copy one image to another.
.help
                         ACOPY

   This PROC was produced as a programming example of memory
```

mapping images into a process's own virtual address space.

However perplexing the above may sound, it is a simple program to use, with only the three parameters.

```
ACOPY IN="FRED" OUT="JIM" SIZE=(1,1,256,256)
```

This will copy the first 256 samples of the first 256 lines from image FRED to image JIM.

```
  ....
.END
```

LISTING 2 - THE LINK PROCEDURE

```
PROCEDURE LINK HELP=*
PARM FILENAME TYPE=(STRING,100)
PARM (DEBUG, MAP, CROSS)    +
  TYPE=(STRING,3) VALID=("YES","NO") DEFAULT="NO"
LOCAL MAPFILE TYPE=FILE INITIAL=-- COUNT=0:1
LOCAL DCLCOM TYPE=(STRING,132) INITIAL="LINK"
BODY
  IF (MAP = "NO")    LET DCLCOM = "&DCLCOM" // "/NOMAP"
  IF (MAP = "YES")   LET DCLCOM = "&DCLCOM" // "/MAP"
  IF (DEBUG = "YES") LET DCLCOM = "&DCLCOM" // "/DEBUG"
  IF (CROSS = "YES") LET DCLCOM = "&DCLCOM" // "/CROSS"
  WRITE "COMMAND IS &DCLCOM"
  DCL &DCLCOM &FILENAME,TAE$BISHOP:USER.OPT/OPTIONS
  IF ($SFI < 0)
    WRITE "Error in linking &FILENAME"
  ELSE
    WRITE "&FILENAME correctly linked"
  END-IF
END-PROC
.title
                    LINK programs with TAE.
.HELP

  This PROC can be used to LINK program modules (object modules
output by any VAX compiler) together with the TAE kernel to
produce executable programs which will run under the TAE command
language.

          ....

.END
```

N87
23166
UNCLAS

N87-23166

POTENTIAL MEDICAL APPLICATIONS OF TAE

J. Ben Fahy, Robert Kaucic, and Yongmin Kim
Department of Electrical Engineering
University of Washington
Seattle, Washington 98195

POTENTIAL MEDICAL APPLICATIONS OF TAE

J. Ben Fahy, Robert Kaucic, and Yongmin Kim

Department of Electrical Engineering
University of Washington
Seattle, Washington 98195

ABSTRACT

In cooperation with scientists in the University of Washington Medical School, we have constructed a microcomputer-based image processing system for quantitative microscopy, called DMD1, for "Digital Microdensitometer #1." In order to make DMD1 transportable to different hosts and image processors, we have been investigating the possibility of rewriting the lower level portions of DMD1 software using TAE libraries and subsystems. If successful, we hope to produce a newer version of DMD1, called DMD2, running on an IBM PC/AT under the SCO XENIX System V operating system, using any of seven target image processors available in our laboratory. Following this implementation, we will transfer copies of the system to other laboratories with biomedical imaging applications. By integrating those applications into DMD2, we hope to eventually expand our system into a low-cost general purpose biomedical imaging workstation. This workstation will be useful not only as a self-contained instrument for clinical or research applications, but also as part of a large scale Digital Imaging Network and Picture Archiving and Communication System, (DIN/PACS). Widespread application of these TAE-based image processing and analysis systems should facilitate software exchange and scientific cooperation not only within the medical community, but between the medical and remote sensing communities as well.

INTRODUCTION

Quantitative microscopy is an important tool for researchers and clinicians in various medical disciplines. It is composed of two quite different methodologies: morphometry, in which spatial properties are measured, and densitometry/fluorometry, which measures mass or activity. The principles which underlie specific techniques of either kind are well understood, and analog instruments ranging in sophistication from conventional microscopes fitted with photomultiplier tubes (PMT) to scanning microdensitometers and flow microfluorometers have emerged. Unfortunately, these instruments tend to be specialized (inflexible), are expensive, and are slow and difficult to interface to computers. As a promising alternative, a digital technique has recently arisen based on interfacing a camera directly to the microscope, and using image processing operations to analyze the resultant digitized images.

In Dec. 1982, we began a joint effort with researchers in the Department of Pathology to develop our own image processing system, and apply it to the measurement of DNA content in hypertensive smooth muscle cells. By Dec. 1984, the prototype system, which we refer to as DMD1 (Digital MicroDensitometer #1) was completed, and it has been heavily utilized in running experiments and analyzing images since that time. New algorithms and features have been continuously incorporated into the software [Nicholls et al., 1985; Vinter et al., 1985; Vinter et al., 1986] with the result that new applications in automated grain counting, immunocytochemistry, and other disciplines can now be developed as natural extensions to our existing system. This expandability is a key advantage of digital methods in quantitative microscopy, since the same basic method can be applied to any number of different applications. Other advantages of DMD1 include improvements in speed (200 cellular DNA measurements per hour vs. approximately 40-50 per hour for conventional analog methods), accuracy, and the possibility of conducting simultaneous analysis of morphometry and densitometry. Comparison of analog and digital systems so far has indicated that digital methods for densitometry/fluorometry are at least as good as the best analog devices [Vinter et al., 1985].

We are now at the stage with our system that it is appropriate to consider making DMD1 (and its successor, DMD2) widely available to other medical researchers by moving the software to a more accessible combination of host and image processor. In its current implementation, DMD1 resides on a custom Motorola MC68010-based host microcomputer and uses a CAT 1600 image processing subsystem (Digital Graphics, Palo Alto, CA). We are in the process of transporting the entire image processing and analysis system to an IBM PC/AT with an ITI (Imaging Technologies Inc., Woburn, MA) FG-100-AT image processor board. Drivers for the image processor under the SCO XENIX System V operating system have been written, and a majority of the DMD1 software has been successfully ported. Because our original system runs UNIX System V and the software was written in a very layered fashion which isolated device-dependent portions of the code, other than writing the new drivers and emulating the CAT image processing functions, transporting the software package has been relatively straightforward.

Now that we have copies of DMD1 on at least two hardware configurations, we face a problem associated with maintaining software compatibility between the two implementations. As new applications are developed on each system, they should be ported over to the other as rapidly as possible. Every time an application takes advantage of the device-dependent features of one system, it will have to be emulated on the other. If a third system is added, then its hardware features will have to be emulated on the other two, and it will in turn have to emulate their special features. Rapidly, as the number of different image processors increases, this emulation strategy is likely to become overwhelmed by the sheer number of combinations which would have to be managed. Therefore, we are seeking to develop alternative transportation strategies now, which will lessen the difficulty in maintaining many different implementations of DMD1 in the future.

The issue of system transportability is being felt not only within our own research group, but is beginning to be recognized within the general medical

image processing community as well. There are many different kinds of image processing systems being used in various clinical and research laboratories. In applications including fluorescent microscopy, microdensitometry, neurological morphometry, autoradiography, and others, hosts range primarily among VAXes, LSIs, PDPs, Eclipses, and PC compatibles [Smith et al., 1985, Ramm et al., 1984, Puls et al., 1986]. The image processors commonly used are manufactured by vendors including Gould, IIS, Grinnel, MATROX, ITI, Datacube, and others . In some cases, such as in our laboratory, special-purpose image processors are designed and built from scratch to meet a particular application need. Other installations may have important peripherals available such as array or signal processors. Because nearly all of these systems have been developed separately, on widely varying combinations of host and image processor, very little constructive sharing has taken place between these research groups. Effectively, this means that each of these efforts has reduplicated the others, leaving little opportunity for the development and implementation of more sophisticated tools.

In what follows we consider the application of TAE to this transportability problem in biomedical image processing, and how we propose to use TAE to make our own system, DMD2, more widely available. We are especially interested in the development of a Biomedical Virtual Image Processor (BVIP), along the lines of the DMS subsystem of TAE. We will discuss briefly the path we intend to take toward incorporating the TAE structure within our system, its potential implementation on a range of seven image processors of widely-varying architecture and capability, and the ramifications of these modifications to the practical possibility of creating a general purpose biomedical imaging workstation. Finally, we will consider the use of such workstations in a large-scale Digital Imaging Network and Picture Archiving & Communication System (DIN/PACS). If we are successful in propagating a TAE-based DMD2 to this extent, it will open up new opportunities for direct and effective software exchange and cooperation.

METHODS

DMD2 is nearly complete, i.e., porting DMD1 to the IBM AT host and ITI image processor. The next step is to implement DMD2 on up to six other image processors in our lab, each of which will be discussed briefly below. For this task, we will replace portions of our system with the TAE Display Management Subsystem (DMS) and refine DMD2 and DMS as necessary to allow the same applications program to run on any of seven different image processors. Following the completion of this task, we will integrate the remainder of DMD2 into the TAE monitor structure, taking advantage of TAE's built-in help, tutor, and other facilities. As it becomes necessary, we will then be in a position to port DMD2 to other operating systems on which TAE is supported, and all new programs written for DMD2 can be created from the beginning within the TAE programming context.

In Figs. 1 and 2 are presented simplified representations of the DMD1 and TAE software architectures, respectively. We will compare the two systems in a top to bottom fashion, noting both the obvious differences and some important

similarities as well. To begin with, in TAE, the top level software module is the TAE monitor, which initiates processes or TCL command language procedures, provides access to the help and menu and tutor facilities, and in many ways can mimic the performance of a generic operating system. There is no equivalent to this module in DMD1. Instead, for most users, a pre-defined sequence of programs is executed, which together perform a full densitometry operation (from digitization through decalibration to cell identification and analysis). This is for the benefit of the medical researchers and technicians, many of whom are unfamiliar with computers to the extent that any deviation from a fixed and rigid pattern of interaction is considered undesirable. For system and application programmers, individual routines may be invoked directly using UNIX. Thus far, this interface has proved adequate, but as the general level of computer expertise within the biomedical community improves, and as DMD2 itself expands to the point where the programmers themselves will require some sort of a help facility, we will need to turn to some sort of a monitor such as is provided by TAE. In fact, as will be discussed below, we eventually plan to implement a version of DMD2 which runs as an application under TAE.

Below the TAE monitor, and as the top layer in our system, reside the applications programs. In DMD2 most of these routines are dedicated to the higher-level functions required for quantitative microscopy. This includes programs for:

- system initialization and calibration
- image digitization and image display (B/W, pseudo color)
- image management (image handling and cataloging)
- interactive device management (tree-structured menu generation)
- histogram generation (whole image or specified region)
- image contrast enhancement (lookup table manipulation)
- algebraic operations (+,-,x,/)
- geometric operations (translation, rotation, roam and zoom)
- 2-D convolution for spatial filtering (with region of interest)
- 2-D Fast Fourier Transform
- various edge enhancement and boundary detection algorithms
- user manipulable cursors for region of interest analysis
- image intensity profile along any specified line
- thresholding for object segmentation and selection
- decalibration to correct uneven illumination in the microscope, and the camera's nonuniform photometric sensitivity
- densitometry operations for quantitative measurements.
- automatic boundary detection algorithms
- automatic morphometry with densitometry
- complexity analysis

The total programming effort for our system in this regard thus far is approximately 4 man years. Each of these routines may be executed as stand-alone procedures or as part of a larger densitometry program chain. Moved into TAE, this chain could be implemented very easily, simply by using a TCL command procedure to invoke the individual applications programs one after the other.

Below the applications layer in TAE is DMS, which provides a device-independent library of standardized image processing functions. It is composed

of an X-layer which is written independent of the particular hardware device being used, and a D-layer, which is device-dependent. In the position corresponding to DMS in DMD2 is the Firmware Extension Layer, created to expand the capabilities of CAT firmware commands. It is important to appreciate that in both systems, these layers (DMS and the Firmware Extension Layer) act as the only interface between the applications programs and the lower-level routines below. This common feature of the two designs is what will make replacement of the Firmware Extension Layer with DMS straightforward.

Below DMS in TAE is the vendor interface layer, which corresponds to the Firmware Interface Layer in DMD2. Each of these layers is designed to contain software functions completely specific to the particular image processor supported. At the lowest level of either software structure are the device drivers, referred to as the Physical Interface Layer in DMD2. As has previously been mentioned, these have already been written for the IBM AT running SCO XENIX System V. Separate drivers are used to perform memory-mapped I/O, manage I/O channels, and service interrupts. In most image processing systems, frame buffers are directly memory-mapped, image processor registers are mapped either through memory or through I/O channels, and interactive devices such as mouses, trackballs, or bit-pads are best implemented using interrupt service routines.

In order to provide a comprehensive and thorough testing ground for the DMS implementation in our system, we intend to create up to seven versions of the device-dependent D-layer, one for each image processor which is available in our lab. These image processors range widely in capability: differences include the number of bits per pixel in the frame buffer (gray-scale resolution), width of the look-up tables, and prescence or absence of graphics overlays, hardware cursor support, display zoom, pan, and scroll, coprocessors, and dedicated image operations hardware, e.g., histogram generation, image arithmetic, and image filtering.

The Digital Graphics Systems CAT-1600 graphics board features real-time digitizing, zoom, pan and scroll, and dedicated graphics and image processing commands implemented in an Intel 8086-based subsystem. Our implementation uses one 512 x 512 x 8-bit bit frame buffer, which is directly memory-mapped in the host address space over an IEEE 696 (S-100) bus. Communication with the host is facilitated by 4 16-bit I/O ports in the IEEE 696 bus: a data port, command port, reset port, and status port. Three independent look-up tables of 8 bits each are assigned to red, green, and blue, enabling pseudocolor options. There is no dedicated hardware support for the cursor, however, a cursor is emulated in the firmware package by overwriting the frame buffer to display the cursor, and restoring the data when the cursor is moved.

The ITI FG-100-AT image processing board resides on the IBM AT bus and contains 16 I/O channel-mapped registers to initiate commands and receive status information. The 512 x 512 x 12-bit frame buffer is directly memory-mapped into the host address space. Three 4096 x 8-bit look-up tables are provided for pseudo color or even true color operation. Like the CAT, real-time digitization, zoom, pan and scroll are supported in hardware. There is no image processor. A feedback loop arithmetic unit is provided, however, whereby 6-bit images may be added, subtracted, multiplied or divided in one frame display time. In our standard operations, we use 8 bits of each pixel for gray level, 3 bits for

graphics, and 1 bit for the cursor. There is no hardware cursor support, which again must be emulated by writing into the frame buffer's cursor plane.

We have had some success already in moving portions of DMD1 to an IBM Professional Graphics Adaptor (PGA) in an IBM AT environment. This raster graphics board consists of a 640 x 480 x 8-bit frame buffer, three 256 x 4-bit look-up tables, and an Intel 8086 based coprocessor which supports three-dimensional graphics and several international standard graphics packages such as the Graphical Kernel System (GKS). Communication between the host and the PGA is accomplished via memory-mapped command, data and error buffers and additional registers. There is no support for hardware zoom, pan, or scroll, and the cursor must be emulated. This system is our lowest capability system.

On the other side of the spectrum is a Gould IP8400 image processing system, which supports many image analysis/processing features in hardware, resulting in high performance. The Gould IP8400 is equipped with three 512 x 512 x 8-bit frame buffers, a video output controller, a pipelined high-throughput digital video processor, and a library of image processing software from Gould. Its host computer is a MicroVAX II, which is networked through DECNET to other computers. In our initial implementation of DMD2, we will run the UNIX operating system, although later versions in which DMD2 runs as an application under TAE could be implemented under VMS.

The TISDB board is a Texas Instruments Software Development Board based on the TMS 34010 Graphics System Processor (GSP) chip. The TISDB has one 512 x 512 x 4-bit frame buffer. It has a look-up table scheme based on their palette chip which allows resetting of the three 16 x 4 bit look-up tables line by line. While this board does not have the gray-scale resolution to be useful for most image processing applications, it has proved a useful tool for gaining familiarity with the powerful GSP chip, and can similarly provide a useful test of DMS. The GSP is a fully programmable 32-bit graphics processor, with special hardware features such as a 256-byte instruction cache and block data move facility, which make it very effective for some image processing operations [Guttag et al., 1986]. Using a C compiler and loader, we have implemented a number of demonstration programs for the GSP, including zooming, convolution, look-up table manipulations, cursor, and menu support. We have also written a resident monitor for the GSP, which loops until given a command from the host to execute a local program. Upon completion, all programs return to the monitor. This interface is implemented both in DOS and XENIX.

We are in the process now of developing a sixth image processor, also based on the GSP, but with much greater capability and gray-scale resolution. This image processor, which we will refer to as UWGSP1, consists of two boards which reside completely within the IBM AT, containing five major sections; the graphics processor, frame buffer, video display, zoom hardware, and signal processor [Chauvin et al., 1986]. It contains four 512 x 512 x 8-bit frame buffers, plus four separate graphics overlay planes, one of which is allocated to the cursor. There are three 4096 x 8-bit look-up tables for red, green, and blue output. Independent vertical and horizontal zoom is supported in hardware, as are pan and scroll. A separate signal processing coprocessor is provided based on the TMS 32020 Digital Signal Processor (DSP) chip.

The last image processor to be used is also in the development stage. It is called UWIP1, and is designed around a special-purpose high speed (30 MBytes/sec) image bus called the IBUS. It features an expandable central frame buffer which currently contains 4 512 x 512 x 12-bit frames. The display processor is based on the Hitachi HD63484 Advanced CRT Controller (ACRTC), and provides independent x and y zoom, three 4096 x 8-bit look up tables for red, green, and blue, hardware support for pan, scroll, cursor movement, and various graphics and annotation functions. Other key modules on the IBUS include a pipelined reconfigurable convolver, arithmetic and logic unit, look-up table transformer and histogram generator, and host interface buffer. Region-of-interest (ROI) operations are implemented in hardware.

To implement DMD2 on such a wide range of image processors, we will begin by writing the DMS D-layer for each processor, using the D-layer provided for the IIS Model 75 as an example. The other portions of DMS, for example the XD, DM, DT, and XL subroutine packages, should port over in a fairly straightforward manner. Because our medical applications do not at this time involve multispectral image analysis, we will delay supporting the image configuration utility, until a specific need for multispectral classification in radiology using images from multiple imaging modalities arises. With DMS available for each several image processors, we will rewrite the Firmware Extension Layer of DMD2 so that the functions called by that layer are implemented using calls to DMS. This will provide the quickest port of DMD2 possible consistent with the TAE architecture, as the Application Layer of DMD2, which was built directly on its Firmware Extension Layer, should be immediately transportable from that point on.

Most DD routines will be fairly routine to implement. Of those routines expected to provide difficulty, most are related either to FORTRAN77 conversions or else to peculiarities of the IIS image processor which made their way into DD. The DDCRDF and DDZMRN routines exemplify special cases in which different image processors may have difficulty in implementing different functions. For example, DDCRDF is for the most part straightforward, except that it allows the possibility of a cursor BLINK attribute. While this could be supported fairly easily on either UWGSP1 or UWIP1, supporting this feature on other processors (without any dedicated cursor hardware) is probably more trouble than it is worth. In the case of the DDZMRN function, the problem is that the function is ambiguous for image processors which support independent x and y zoom, or arbitrary-size zoom, such as UWGSP1 or the TISDB.

It is important to note that DD in its current implementation leaves out many hardware features of our in-house image processors which can be quite important to the efficiency of the running system. For example, the Device Characteristics Mask of the DMS Display Device Table contains only 8 hardware characteristics which are checked for existence thus far: hardware zoom, histogram generator, split screen, image shift, scale on input, look-up table bypass, alpha generator, and keypad buttons. Clearly, these features have been singled out with a specific image processor in mind. But almost certainly in our implementation, in order to achieve maximum efficiency, we will also have to include characteristic flags for:

- independent X and Y zoom (ITI, UWIP1, UWGSP1, TISDB)
- hardware convolver (UWIP1, UWGSP1)
- array or signal processor (UWGSP1)
- arithmetic unit (ITI, UWIP1, UWGSP1, TISDB)

These capabilities will have to be incorporated into the system model of image processing characteristics in order to fully take advantage of the hardware available. In turn, new routines should be added to DD and XD to allow these functions to be callable from applications programs. Another important consideration which DMS currently seems to leave unresolved is the nature of communication with the image processor. Does it reside on a separate bus? Are frame buffers directly memory-mapped, which greatly simplifies image loading, or is a more complicated interface required? Should interrupt routines be used as an integral part of the communication strategy, or is a polling or master/slave communication model sufficient?

With the wide range of image processors available in our laboratory to experiment on, we expect that we will experience a great many difficulties in adapting a consistent DMS interface to each image processor. On the other hand, it is our hope that this experience will prove very profitable in that solutions to these problems inevitably will be worked out, and a more sophisticated model of image processor capability than is presently outlined in DMS may result. It should be noted that as image processing hardware continues to improve over time, the model will continually have to be updated to correspond to the new state of the art: the creation of a "virtual image processor," then, which is the fundamental goal of DMS, will indefinitely remain a very dynamic process. It is our feeling that the virtual processor model will be kept best up to date by continually subjecting the model to new and widely-varying capability image processors, such as we will be attempting to do in the case of our widespread implementation of DMD2.

DISCUSSION

The previous section has outlined the method by which the DMS subsystem of TAE may be used to aid in the portability of our digital microdensitometer to a wide variety of image processors. The immediate beneficial effects of such an exercise are (1) to achieve the port itself, and (2) to improve the virtual image processor model of DMS to include a more comprehensive list of image processor functions. In addition to these immediate benefits, however, there are other longer-term advantages which are also important to consider.

Either after or in parallel with the implementation of DMS in DMD2, we will turn to the inclusion of the DMD2 application programs within the general framework of TAE. This would most likely be attempted first for the IBM AT/ITI-based system. Besides creating TAE format help files for the programs, some restructuring might prove necessary because in DMD2, many function options are currently selected via the cursor on a display-based menu, as opposed to on a separate terminal. It should be noted, however, that it is likely that such restructuring will only have to be carried out once, since after menus are moved to a separate terminal, the rest of the TAE application program interaction

should be dependent only on the IBM AT host, and not on the particular image processor chosen.

With DMD2 available as an application package under TAE, it should become possible to port DMD2 to other operating systems on which TAE is supported, e.g., the VMS operating system available on our MicroVAX II. We will proceed to integrate DMD2 with other biomedical image processing systems and applications so as to fill out its overall functionality, approaching the creation of a general purpose biomedical imaging workstation. Since each installation will be based on the TAE structure and libraries, it should be much easier to share software between users of this system, as well as with the investigators of completely different fields of endeavor in the NASA remote sensing community.

As a final example of the utility of this approach, we consider the application of the resultant workstation to a large scale Digital Imaging Network and Picture Archiving and Communications System (DIN/PACS) [Alzner et al., 1986; Parrish et al., 1986]. The DIN/PACS is being be designed to aid radiologists in interpreting images and associated data for medical diagnosis, and will provide for:

- image capture by the system
- storage of images
- image retrieval for diagnostic and display purposes
- image manipulation, arrangement and enhancement during diagnostic viewing
- entry and retrieval of clinical data
- diagnostic report generation
- indexed retrieval and statistical analysis for research purposes
- medical education activities
- radiology department management

The fundamental advantage of DIN/PACS over the present image management approach used in a typical radiology department is that it will allow physicians and radiologists to combine and analyze simultaneously all available information on a patient, especially including internal images produced by such imaging modalities as Computer Aided Tomography (CAT), Magnetic Resonance Imaging (MRI), Digital Subtraction Angiography (DSA), Positron Emission Tomography (PET), Ultrasound, and digitized X-Ray film. Each modality presents a different view of the internal state of the body, and by combining the information obtained from all modalities, it may be possible to significantly increase the effectiveness of noninvasive radiological diagnosis.

Because of the wide ranges of image processing capability which is expected to be used in DIN/PACS, it would be advantageous if the software used to develop and manage DIN/PACS image processors contained an effective virtual image processor model such as DMS may eventually provide. Further, it would be highly desirable to have some means of facilitating communication, exchange of software and ideas, and general cooperation between the propagators of DIN/PACS and the general image processing community. Therefore, if we are successful in developing the biomedical imaging workstation in a timely fashion, we will attempt to incorporate it into DIN/PACS, providing a TAE link between medical

imaging and the remote sensing communities.

SUMMARY

Biomedical image processing applications have been developed relatively recently compared to applications in remote sensing. As a consequence, many biomedical laboratories and clinical installations are just now beginning to meet head-on the problem of software portability which TAE was designed to combat. In our own laboratory, we have developed a microcomputer-based image processing system for quantitative microscopy, which we would like to port to other hosts and image processors in order to expand its capabilities to that of a biomedical imaging workstation. Because of the wide range of image processors available to us, we have decided to replace the image processor interface portion of our system with the DMS subsystem, thereby forcing the development of a Biomedical Virtual Image Processor which should greatly aid portability within the biomedical community. As part of this process, DMS will be severely tested, refined and enhanced. As the workstation develops, it will be possible to incorporate it within a large scale DIN/PACS. By using TAE as an integral part of these medical image processing systems, general software exchange and scientific cooperation will be facilitated between the medical and remote sensing image processing communities.

REFERENCES

Alzner, E., and Murphy, L., "The ACR-NEMA digital imaging and communications standard: evolution, overview and implementation considerations, Proceedings of SPIE, Vol. 626, pp. 506-510, 1986

Chauvin, J. W., Steiner, A. R., and Kim, Y., "Image processing with the TMS34010 Graphics System Processor," Digital Design, submitted, 1986.

Guttag, K., Van Aken, J., and Asai, M., "Requirements for a VLSI Graphics Processor," IEEE Comput. Graphics Applica., Vol. 6, pp. 32-47, Jan. 1986.

Nicholls, W. H., Kim, Y., Schwartz, S. M. and Vinter D. W., "Development of a microcomputer-based microdensitometer," International Journal of Microcomputer Applications, accepted, 1986.

Parish, D. M., Creasy, J. L., Thompson, B. G., Rogers, D. C., Johnston, E. R., Perry, J. R., Staab, E. V., "Functional requirements for interfacing PACS to RIS," Proceedings of SPIE, Vol. 626, pp. 597-602, 1986

Puls, J. H., Bibbo, M., Dytch, H. E., Bartels, P. H. and Wied, G. L., "MicroTICAS: The design of an inexpensive video-based microphotometer/computer system for DNA ploidy studies," Anal. Quant. Cytology Histology, Vol. 8, pp. 1-7, 1986.

Ramm, P., Kulick, J. H., Stryker, M. P. and Frost, B. J., "Video and scanning microdensitometer-based imaging systems in autoradiographic densitometry," J. Neurosci. Methods, Vol. 11, pp. 89-100, 1984.

Smith, L. C., Jericevic, Z., Benson, D. M., and Bryan, J., "Digital imaging fluorescence microscopy," IEEE Frontiers Eng. & Comp. Health Care, Vol. 7, pp. 967-970, 1985.

Vinter, D. W., Nicholls, W. H., Kim, Y. and Schwartz, S. M., "Development of a biomedical image processing computer system," IEEE Frontiers Eng. & Comp. Health Care, Vol. 7, pp. 996-1000, 1985.

Vinter, D. W., Kim, Y. and Schwartz, S. M., "Assessment of digital microdensitometry for histochemical applications: comparison with fluorescent methods for DNA determination of vessel wall cells," Lab. Invest., submitted, 1986.

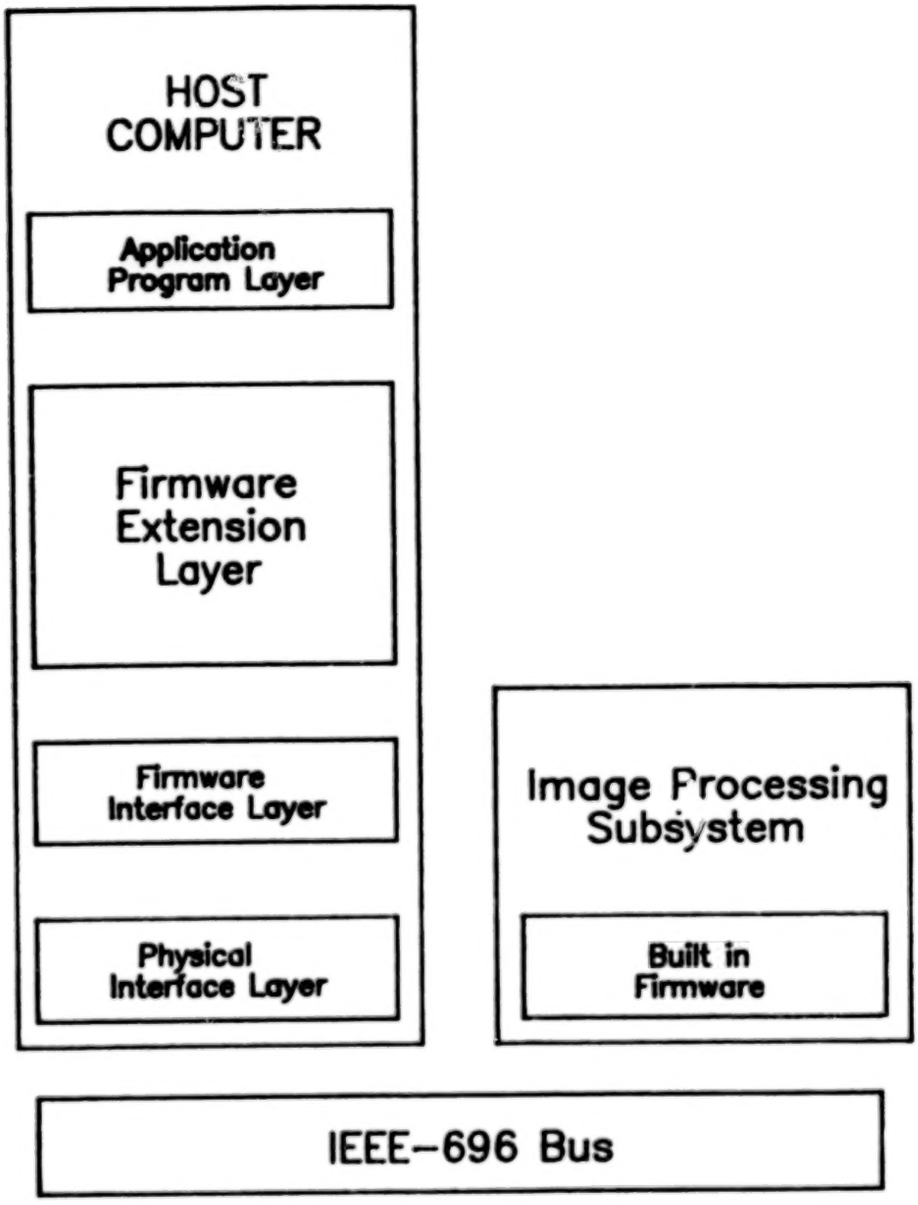


Figure 1. The layered interface software design in DMD1.

C-3

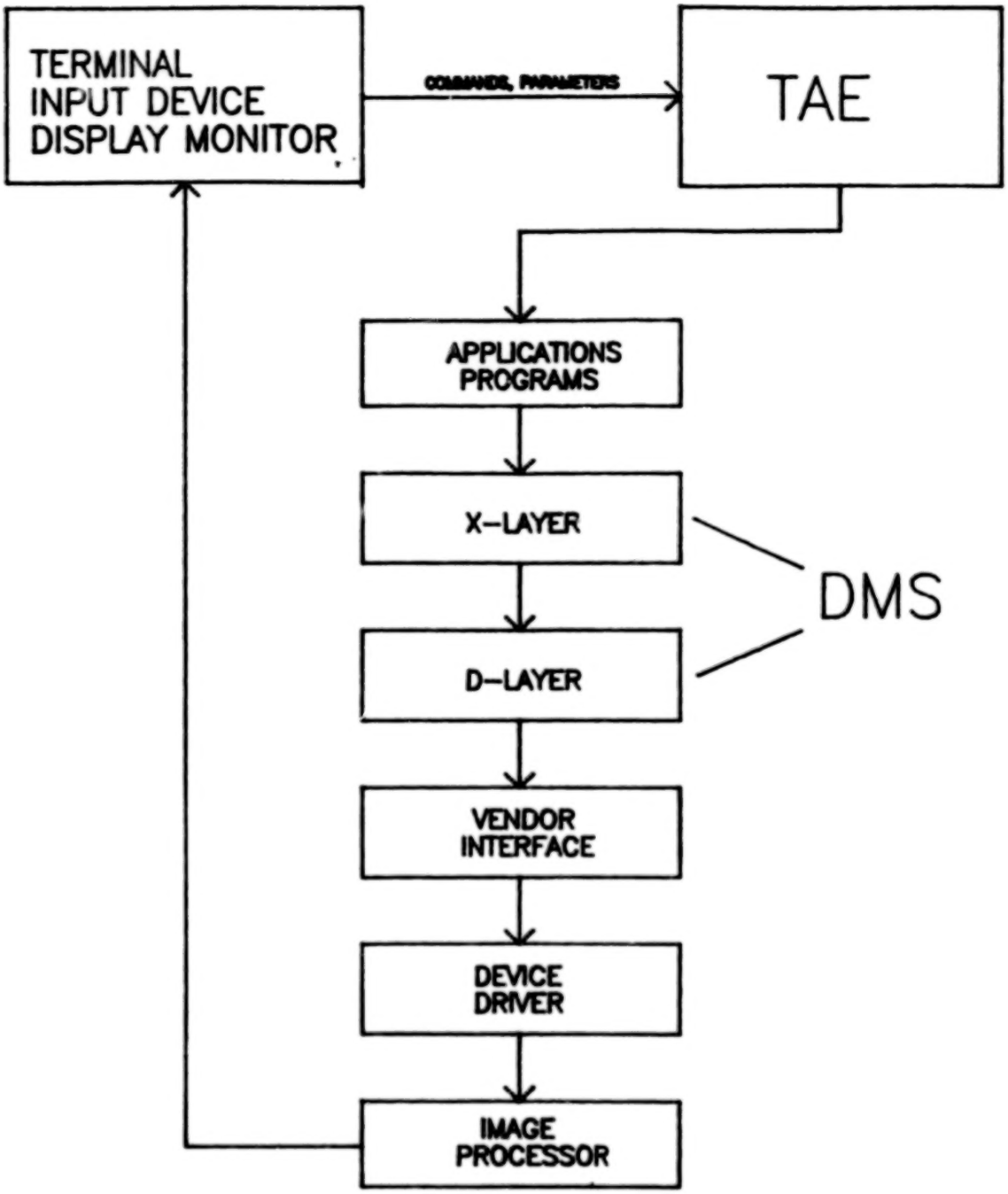


Figure 2. Simplified block diagram of TAE and DMS.

N87
23167
UNCLAS

N87-23167

NAVAL APPLICATIONS OF A TAE-DERIVED EXECUTIVE

Michael Guberek, Stephen Borders and Serge Masse
Global Imaging, Inc.

NAVAL APPLICATIONS OF A TAE-DERIVED EXECUTIVE

Michael Guberek, Stephen Borders and Serge Masse
Global Imaging, Inc.

Global Imaging introduced an interactive image processing system in 1985, featuring the Global Applications Executive (GAE) which is a modified TAE environment. The executive plus a large variety of image processing functions, known commercially as the System 9000, are designed to operate on the Hewlett-Packard family of Unix computers. Beacause the US Navy has chosen the Hewlett-Packard as its standard desktop computer (NSDTC), the System 9000 has found easy acceptance for naval image processing applications. In 1986, Global has installed its systems at the US Naval Academy, the Naval Research Laboratory and the Naval Oceanographic Facility. Other naval installations are currently considering acquisition of this capability.

The Department of Oceanography at the Naval Academy, Annapolis, Maryland, has installed an NSDTC with an image processing upgrade provided by Global Imaging. This interactive digital image processing workstation is used by the midshipmen and staff for training and research in remote sensing oceanography. The turn-key system provides the capability to process imagery from commonly used earth observation spacecraft, in conjunction with in situ data sets. This marks the first time an undergraduate oceanographic curriculum offers such an advanced capability.

The Acoustics Group at the Naval Research Laboratory, Washington, DC has acquired its first System 9000 to interactively process ocean acoustic data gathered by shipboard sensors. The vertical acoustic profiles had been processed up to now by a laborious batch process. The ease of user interaction will permit faster analysis of the raw data.

Several naval installations are currently working on the Tactical Environmental Support System (TESS). This shipboard computer system is tasked with consolidating oceanic and atmospheric information to aid tactical decisions. The first and second generations of TESS are based on the NSDTC hardware, while the third generation hardware is still under consideration. The Naval Oceanographic Facility in Bay St. Louis, Mississipi has recently acquired a System 9000 to provide a TESS 2 prototype with image processing capabilities. This will permit merging of conventional data with polar orbiting spacecraft imagery.

Global Imaging is proposing to utilize GAE as the user interface for the third generation of TESS. In this system, each shipboard computer requires five workstations, four of which must be capable of interactive image processing. The imagery will be acquired in real-time from the NOAA, DMSP and NROSS polar orbiting satellites. In addition, a vast array of climatological, geophysical and recent environmental data must be analyzed to provide tactical decision aids for ship and aircraft operations at sea. This complex processing environment can be greatly simplified by providing the operators with the tutor, menu and help features of GAE.

Hardware

The current System 9000 is based on the Hewlett-Packard Series 300 and 500 high-performance 32-bit processor (CPU), with a direct address range of 500 Megabytes. A separate input/output processor (IOP) frees the CPU from functions associated with direct memory access by peripherals such as disk drives and displays. The modular design of this computer permits multiple CPUs and IOPs to reside on the same bus to provide increased performance when necessary. The computer configuration, includes a central processor unit (CPU), 4 Megabytes of solid state memory, and one IOP capable of handling direct memory access (DMA) transactions on 8 independent channels. The new Series 800 processor, based on reduced instruction set (RISC) architecture, delivers 4.5 MIP performance while retaining software compatibility with the previous generation processors.

Mass storage is provided by fixed or removeable media disk drives. The Hewlett-Packard 7900 series drives combine Winchester technology, a separate HP 9144 cartridge tape drive supplying cost effective backup and user I/O, and a sophisticated microprocessor-based controller to manage both storage components, all integrated into a single compact package. A 16 ms. average access time and a data transfer rate of up to 1 Megabyte per second make these drives among the highest performing mass storage devices available.

The Metheus Omega 3610 display controller is used to drive the color CRT display. The controller memory is configured to hold 1024X1280X32-bit images. These are displayed at 60 Hz non-interlaced refresh rate using bright color monitors. The custom bit-slice processor contained in the Omega 3610, with a cycle time of 167 nanoseconds, can flash-fill rectangles at 160 million pixels per second.

The display controller communicates with the HP 9000 via a 16-bit parallel interface, using a simple byte-oriented protocol. The command structure minimizes the the number of bytes which must be transferred from the host to perform time-critical tasks. A comprehensive instruction set (Table 1.) simplifies the design of host computer software and contributes to efficient operation.

The programmable cursor allows any size cross-hair cursor to be specified. A set of 122 characters (8x16 pixels) is resident in ROM and is loaded into a special character RAM at power up. Space is provided for asecond character set as well. Either character set may be chosen by the user. Instructions are provided for scaling character size and specifying character spacing.

Software

The HP-UX Operating system, a licensed and supported version of AT&T's System V, provides a productive environment for the development and execution of image processing software. Programmers andusers may easily share and reuse files and tools without sacrificing system security and reliability. The HP-UX operating system features multi-user capability, multitasking capability, virtual memory for code and data, and engineering tools such as data base management and data communications.

The workstation provides the Global Applications Executive, which standardizes the link between the user and applications programs under the UNIX operating system,. The user can operate the system in three modes. In the menu mode, the user is asked to make a selection from a list of menus and applications. In the command mode, the user communicates with the system via simple English-like commands. Finally, in tutor mode, the user is prompted for all parameters which must be supplied to a program. In both the command and tutor modes, the applications executive checks that all parameters entered by the user lie within the correct range. In command mode, if any parameter lies outside a valid range, the user is asked to re-enter the command; in tutor mode, he is re-prompted for the parameter. Parameters may be optional or mandatory. Optional parameters not specified by the user are assigned default values by the executive. Help files, which display information on the operation of the menu, command, and tutor mode as well as on the operation of all application functions, are available.

A Programmer's Subroutine Package was provided to accelerate applications software development. This package contains FORTRAN callable I/O routines for reading data from existing images as well as creating new images. Images created using this subroutine package are automatically cataloged and can contain any number of lines and samples with up to 64 bands. Image data can be any one of five types ranging from one byte per pixel, to 64-bit real values per pixel.

A sophisticated Graphics Software Package (GSP) has been written to support the Omega 500 Display Controller. At logon, the GSP assigns a display, if one is available to each user. The user can operate in one of three modes. In the first mode, the display is divided into four 8-bit image planes; in the second mode, three 8-bit image planes and one graphics overlays are enabled; and in the third mode, two 8-bit image planes and two graphics overlays are used. The GSP keeps track of the current display mode, active image and graphic overlay planes, and active monochrome or pseudocolor look-up tables. Up to 32 pseudocolor and 32 monochrome look-up tables can be stored and easily recalled.

The applications software includes programs to perform geometric correction, earth location. and registration of remotely sensed data. These programs handle imagery from the Advanced Very High Resolution Radiometer (AVHRR), the Coastal Zone Color Scanner (CZCS), the Multispectral Scanner (MSS), the Scanning Multichannel Microwave Radiometer (SMMR), and the Visual and Infrared Spin Scan Radiometer (VISSR). Other programs permit displaying monochrome and true-color images. Line graphics can be overlayed onto the displayed image in different colors. Interactive manipulation of these images is possible via a digital tablet provided. Interactive function include panning, histogram normalization and pseudocolor manipulation.

N87
23168
UNCLAS

N87-23168

AN OVERVIEW OF THE LAND ANALYSIS SYSTEM (LAS)

Yun-Chi Lu
NASA/GSFC

LAND ANALYSIS SYSTEM

Yun-Chi Lu

Code 636

Information Analysis Facility

Goddard Image and Information Analysis Center
Space Data and Computing Division
NASA/Goddard Space Flight Center

AGENDA

- History
- Development Methodology
- Major Hardware and Software Components
- Hardware Configuration
- Independent Audit--Evaluation Criteria and Approach
- Desired Enhancements
- Configuration Control Board
- Dissemination of LAS

LAS
Land Analysis System

HISTORY

- Lansat-D Assessment System--1980
- Landsat-D Assessment System--1981
- Land Analysis System (LAS)--August 1983
- Independent Audit Started--Feburary 1984
- Outside User Contribution (EROS Data Center)--June 1984
- LAS Configuration Control Board--June 1985
- LAS Version 3.1 Release--August 1985
- LAS Available Through COSMIC--July 1986

LAS
Land Analysis System

REQUIREMENTS

- User Interface (TAE)
- Functional Capabilities
- System Support Services
- Documentation
- System Performance

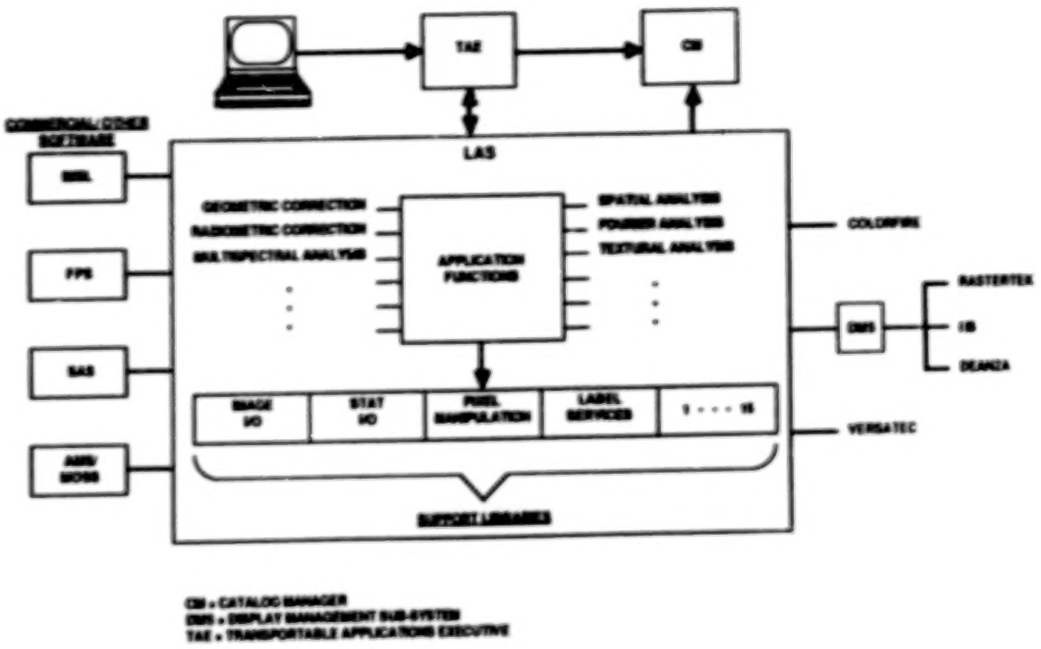
DEVELOPMENT METHODOLOGY

- Define Requirements
- Design and Review
- Implementation
- Unit Testing
- Integration and System Testing
- Acceptance Testing
- Configuration Control
- Independent Auditing

DESIGN ELEMENTS

- Batch and Interactive Processing
- Flexible User-System Interface
- Extensive Session History
- Automatic Cataloging of Data Sets
- Menu and Command Mode Processing
- Multi-level Help File for All Processing Functions

THE LAND ANALYSIS SYSTEM



CM = CATALOG MANAGER
DMS = DISPLAY MANAGEMENT SUB-SYSTEM
TAE = TRANSPORTABLE APPLICATIONS EXECUTIVE

SYSTEM SUPPORT SERVICES

- Transportable Applications Executive (TAE)
  - User Friendly Interface
  - Online Help
- Catalog Manager
  - Meaningful Names for Images and Data Files
  - Archival and Retrieval Functions
- History Files
  - Complete Processing History Information for all Images
- Applications Services
  - Assembly Language Codes to Help Programmers (Image I/O, Statistics I/O, and Pixel Manipulation
- Session Logging
- Ancillary Data Processing
  - TM HAAT Files
  - Statistics
  - Image Registration Points

LAS HARDWARE SYSTEM SUMMARY

VAX-11/780 ---> CLUSTER

8 MBYTES MEMORY

AP180V ARRAY PROCESSOR

8 RP06 MOUNTABLE DISKS (@176 MBYTES)

3 RA81 FIXED DISKS (@450 MBYTES)

3 (2) 6250 BPI TAPE DRIVES

3 (2) HAZELTINE IMAGE TERMINALS

2 IIS MODEL 75 IMAGE TERMINALS

FILM RECORDERS

- DICOMED
- OPTRONICS L5500 B&W
- MATRIX CAMERA
- COLORFIRE 240

LAS
Land Analysis System

USER INTERFACE

- **The LAS is integrated under the Transportable Applications Executive (TAE).**
  - **Human Engineered User Interface**
  - **Extensive On-Line Multi-Level Help Files**
  - **Menu and Command Mode Processing**
  - **Tutoring Capability**
  - **Parameter Save File**
  - **Programmer Interface**

FUNCTIONAL CAPABILITIES

- A total of 224 applications programs were developed in response to users' requirements.
  - Arithmetic and Logical Functions
  - Data Transfer Functions
  - File Management Functions
  - Fourier and Complex Image Functions
  - Geometric Transformation Functions
  - Hard Copy and Terminal Listing Functions
  - Image Restoration
  - Intensity Transformation Functions
  - Multispectral Processing Functions
  - Spatial Processing Functions
  - Statistics and Sampling Functions
  - Miscellaneous Functions

```
Menu:   "ROOT", library "TAE $MENU"

        ************************************************************
        *                                                          *
        *         LAND  ANALYSIS  SYSTEM  -  Version 3.1B          *
        *                                                          *
        *                       AUGUST, 1985                       *
        ************************************************************

1)  System I/O Functions Menu
2)  Applications Functions Menu
3)  Image Display Functions Menu
4)  Utility Functions Menu
5)  Catalog Manager Functions Menu
6)  TAE Session Log Functions Menu
7)  General Information menu



Enter:  selection number, HELP, BACK, TOP, MENU, COMMAND, or LOGOFF.
```

```
Menu:   "APPLIC", library "LAS$MENU:"

1)  Arithmetic Functions Menu

2)  Classification Functions Menu

3)  Fourier Transform Functions Menu

4)  Geometric Rectification Functions Menu

5)  Logical Functions Menu

6)  Radiometric Correction Functions Menu

7)  Sampling Functions Menu

8)  Spatial Functions Menu

9)  Applications Utility Functions Menu


Enter:  selection number, HELP, BACK, TOP, MENU, COMMAND, or LOGOFF.
```

```
Menu:   "CLASS", library "LAS$MENU:"

              **********************************
              *  CLASSIFICATION FUNCTIONS      *
              **********************************

1)  Supervised Classification Functions
2)  Unsupervised Classification Functions
3)  Classification Utility Functions









Enter:  selection number, HELP, BACK, TOP, MENU, COMMAND, or LOGOFF.
```

```
Menu:   "UNSUPER", library "LAS$MENU:"

          ********************************************
          *  UNSUPERVISED CLASSIFICATION FUNCTIONS   *
          ********************************************

 1)  Linear Discriminant Analysis                      (DISCRIM)
 2)  Clustering via Histogram                          (HINDU)
 3)  Clustering via Cluster Distances                  (ISOCLASS)
 4)  Performs a clustering classification              (KMEANS)
 5)  Apply polygonal mask to an image                  (MASK)
 6)  Combine level 1 and level 2 classifications      (SPECCOMB)
 7)  Stratifies a multi-spectral image                 (SPECSTRT)




Enter:  selection number, HELP, BACK, TOP, MENU, COMMAND, or LOGOFF.
```

```
Tutor: proc "ISOCLASS", library "LAS$APPL"                          Pg 1+

Performs an unsupervised classification using an ISODATA algorithm

  parm          description                                  value
  ----          -----------                                  -----

IN              (Required)
                Input image.


OUT             Output classified image.









Enter: parm=value,HELP,PAGE,QUALIFY,SHOW,RUN,EXIT,SAVE,RESTORE; RETURN to page
```

```
Tutor: proc "ISOCLASS", library "LAS$APPL"                    Pg 3+

Performs an unsupervised classification using an ISODATA algorithm

  parm        description                              value
  ----        -----------                              -----

SPOUT         Output statistics file

MAXIT         Maximum number of                          2
              iterations

DLMIN         Threshold for combining                  3.2
              clusters

Enter: parm=value,HELP,PAGE,QUALIFY,SHOW,RUN,EXIT,SAVE,RESTORE; RETURN to page
```

```
Help:  parameter "MAXIT", proc "ISOCLASS"                     Pg 1+

MAXIT specifies the maximum number of clustering iterations.

With each iteration, ISOCLASS passes through the input data and assigns
pixels to clusters using either a split or combine operation.  Program
execution will terminate once MAXIT iterations have occurred, or the user
may interrupt processing by using the 'VIEW' parameter.  See 'VIEW'.

Enter EXIT or PAGE n (or press RETURN for next page)
```

FUNCTIONAL CAPABILITIES--DISPLAY

- Interim Solution--Bridge Between TAE and IIS CI
- Permanent Solution--Available in December 1986
  Display Management Subsystem (DMS)
  - IIS
  - DeAnza
  - Raster Technologies
  - Adage

INDEPENDENT AUDIT--APPROACH

- Module Test = A total of 224 modules
- Macro-Module (Scenario) Evaluation

Number	Macro-module Descriptions
13	Data transfer
7	Preprocessing
10	Geographic image registration
9	Data transformation
7	Creation of raster images from digitized map data
13	Supervised classification
8	Unsupervised classification
16	Spatial and frequency feature extraction
2	SAS interface testing
24	Display subsystem
2	Catalog manager and tape library

SYSTEM PERFORMANCE

- Speed = CPU and I/O
- Accuracy = Validity of Results

CLASSIFICATION OF HUNTSVILLE, ALABAMA USING THE TASSELED CAP TRANSFORMATION

LAS FUNCTION	
CCTTIPSP COPY FACTOR SCALE	PREPROCESSING
TIESELECT REGISTER COPY	REGISTRATION
ISOCLASS COLOR RENUMBER MASKSTAT BAYES CONTABLE	CLASSIFICATION
COLOR LUTSAV GROUP CFIRE	DISPLAY

DESIRED ENHANCEMENTS

- Display
- Reformat Session History
- Catalog Manager
- AP Improvement
- AMS/MOSS Interface With LAS
- UNIX Conversion
- Porting to Microcomputer

CONFIGURATION CONTROL BOARD (CCB)--JUNE 1985

Board Members:

Lyn Oleson:	EROS Data Center/Computer Services Branch
Bruce Quirk:	EROS Data Center/Applications Branch
Stephen Wharton:	GSFC/Laboratory for Terrestrial Physics
Yun-Chi Lu:	GSFC/Space Data and Computing Division

DOCUMENTATION

- Applications Programmer's Guide
- LAS User's Manual (on-line and off-line)
- LAS Installation Guide

DISSEMINATION OF LAS

- Documentation/Information Through User Support Office [GSFC/(301) 286-6034]
- Software Through COSMIC, University of Georgia, Athens, GA 30601

N87
23169
UNCLAS

N87-23169

DEVELOPMENT OF LAND ANALYSIS SYSTEM DISPLAY MODULES

Douglas Gordon, Douglas Hollaren, Laurie Huewe
TGS Technology, Inc.
EROS Data Center, Sioux Falls, SD

Development of Land Analysis System Display Modules

Douglas Gordon
Douglas Hollaren
Laurie Huewe

TGS Technology, Inc.
EROS Data Center Sioux Falls, SD

I. Introduction

The LAS display modules were developed to allow a user to interactively display, manipulate, and store image and image related data. To help accomplish this task, these modules utilize the Transportable Applications Executive and the Display Management System software to interact with the user and the display device. Figure 1 shows how these components relate to one another and the shaded box is where the major focus of this discussion will be. First, the basic characteristics of a display will be outlined; some of the major modifications and additions made to the display management software will be discussed next; finally, all available LAS display modules along with a short description of each will be listed.

II. Basic Display Characteristics

The LAS display modules do not interact with one specific type of device, but they do assume the display has several basic characteristics. They are: (1) a set of memory planes for storing image data, (2) a set of graphics planes for storing graphics data such as text, linework, histograms and tick marks, (3) a look-up table for mapping image data, and (4) a pointing device for user interaction with the display.

A display can be thought of as a stack of memory and graphics planes (figure 2). Memory planes can be described as two-dimensional planes with each pixel in the memory plane having an intensity value that ranges from 0 through 255. Graphics planes are similiar in organization to memory planes except that each pixel of a graphics plane does not hold a value between 0 and 255 but rather a value of either 1 or 0 representing either an "on" or "off" state. Graphics planes are usually used as an overlay to an already displayed image and may be displayed in a number of colors depending on the display.

A look-up(LUT) table allows a user to alter or map the relationship between the values in the memory planes and the displayed intensities (figure 3). A look-up table can be thought of as a list of "from" and "to" translations such that each possible input data value (0 through 255) is an index into a table of output intensity values. For example, if the first entry in the LUT is 100, every pixel in the memory that has a value of 0 will be displayed as though it had an intensity of 100. The original image data remains unchanged but is displayed with different intensities. Each LUT has three components--one for the red band or component of the image, one for the green component, and one for the blue component.

The fourth characteristic, a pointing device, is usually available with a display and includes such devices as a trackball, joystick, or mouse. These devices provide capabilities such as cursor movement. Many of the LAS display modules also make use of the function buttons found on these devices.

III. Modifications and Additions to the Display Management System

In designing and developing the LAS display modules, it was discovered that several modifications to the prototype DMS that we were working with were necessary to meet all the user-defined requirements. These changes were adopted by the Goddard Space Flight Center and have been incorporated in the DMS presently being distributed. One such modification involved changes to several of the DMS tables which store device and image tracking information. Most of these modifications consisted of moving fields from one table to another or adding fields to a table. The first table that needed modification was the display device table which holds device specific information (figure 4). Fields were added to store cursor information which includes the active cursor number, the cursor shape, size, color and blink rate. A graphics plane mask was also added to store the graphics planes which are currently on. The display memory table which holds information about each memory plane in a device was modified next (figure 5). Several fields were moved from this table to the image configuration table. This was to allow each defined image to have its own shift, zoom, memory window, and source file name rather than linking these attributes to a specific memory plane. The last table to be modified was the image configuration table which holds information about each defined image (figure 6). The image configuration table was renamed to the display parameter table to clear up some user confusion and the image group information was removed. A field was also added to keep track of LUT table information, to store a graphics active record list name, and to store the LAS file window and band numbers.

The second modification involved coordinate conversions. These routines caused one of the largest problems encountered in LAS display module development. The LAS display modules use three different coordinate systems to reference the spatial locations of the image (figure 7): (1) file coordinates which represent the line and sample location from the upper-left corner of the original LAS image as it resides on disk, (2) image coordinates which represent the line and sample location from the upper-left corner of the image as it resides in the memory plane, and (3) screen coordinates which represent the line and sample location from the upper-left corner of the display screen. Because of the different coordinate systems, it was necessary to write coordinate conversion routines. Differences in display origins, differences in the manner that displays handle zooming and shifting of memory planes, and differences in how pointing devices return cursor coordinates made it difficult to integrate the conversion routines.

The method of shift calculation was also modified and was made a device dependent calculation. Originally, the same shift calculation was used on all types of devices. This caused problems with devices that handled shifting differently. For example, the Raster Technologies Model One/25 display only allows shift values in multiples of 4 in the X direction and multiples of 2 in the Y direction but the Deanza IP8500 display does not impose this restriction. Therefore, a different shift calculation routine was needed for each.

In addition to the modifications made to DMS, several new utilities were also written to save, recall, and access information from three types of image related files.

The first type, the display parameter file, is a disk file that stores the parameters necessary to recreate a view of an image. This information includes the zoom and shift factors applied to the image, the LAS image name, the LAS image window, the LAS image bands, and the look-up table information.

The second type of image related files are the graphics overlay files which are disk files that contain information necessary to display graphics data along with descriptive attributes for the data. There are four types of graphics overlay files: files that store point data, files that store line data, files that store polygon data, and files that store annotation data. In conjunction with these modules, routines were also written to clip the graphics data and to automate label placement.

The last image related file is the active record list. This session temporary disk file contains the name of the graphics overlay file it references, pointers to the currently active records of the graphics overlay file, and graphics plane masks to indicate which graphics planes the data is currently displayed in. These active record lists allow a user to display or manipulate a defined subset of graphics overlay data which helps speed processing time.

Routines are also currently in development to convert the graphics overlay data to formats useable by other software packages.

IV. LAS Display Applications

The LAS display modules themselves have been grouped into five categories. Color display modules provide status information, allow allocation and deallocation of the display, allow loading and manipulation of images in the display memory planes, and saving images from the memory planes back to disk. Mapping modules apply, save, and restore intensity and pseudo color mappings. Graphics overlay modules create, save, modify, and restore text and linework and generate histograms and tic marks. Arithmetic modules generate output images by performing arithmetic, logic, rotation, and convolution operations on images, and cursor modules define and turn a cursor on and off and determine cursor locations and intensity values.

The following is a list of all the LAS display modules available at the EROS Data Center along with a short description of each. These modules are operational on a Deanza IP8500 display running on a VAX 11/780 with a VMS operating system and on a Raster Technologies Model One/25 display running on a SUN Microsystems workstation with a UNIX operating system at the EROS Data Center. A subset of these modules are also operational on a I2S Model 75 at the Goddard Space Flight Center. As for the future of the LAS display module development, there are approximately eight modules left to develop and/or enhance. Some of these enhancements involve taking advantage of display hardware characteristics to improve performance. There are also plans to implement these modules on an I2S IVAS display at the Western Mapping Center in Menlo Park, California, and on a Deanza IP8500 display at the EROS field office in Anchorage, Alaska.

Module	Description
ADJUST	Allows the user to interactively adjust the brightness and/or the contrast of a displayed image with a linear mapping through movement of the pointing device. Horizontal movement adjusts brightness, and vertical movement adjusts contrast.
ALLOC	Allocates a display for a user and optionally initializes that display following allocation. A display must be allocated before executing any other display functions with the exception of DSTAT and some of the graphics overlay functions.
ARITH	Allows the user to perform several arithmetic operations on images. ARITH-ADD allows the user to add two images; ARITH-SUB allows subtraction of one image from another; ARITH-MULT allows multiplication of two images; and ARITH-DIV allows division of one image by another. Note that there must be a memory plane available for storing the resulting image data. All arithmetic operations involve only two images.
CPYGOF	Subcommand -ARL copies records from an active record list into a specified graphics overlay file. Subcommand -GOF copies records from a specified graphics overlay file into another graphics overlay file. Deleted records will not be copied. Subcommand -CLEAN removes deleted records from the specified graphics overlay file.
CURPOS	Displays the line and sample values in file, image, screen, or memory coordinates; also displays the image intensity value(s) at the cursor position for each image band currently being displayed. The intensity values for the image being viewed are displayed in RGB order. The default is to display the file line and sample coordinates along with the image intensity values.
CURSOR	Turns the cursor on or off.
DALLOC	Deallocates the display allocated by a user.
DEFATT	Defines an attribute name or list of names for a graphics overlay file (GOF). A maximum of 35 attribute names may be defined for a single GOF type.
DEFCUR	Defines a cursor. DEFCUR allows the user to define the shape, size, color, and blink rate of a cursor. The defined cursor will be turned on at the center of the screen.

DELATT	Deletes an attribute name or list of names from a graphics overlay file (GOF). A maximum of 35 attribute names may be deleted from a single GOF.
DELDPF	Deletes the specified display parameter file (DPF) entry from the DPF associated with the specified image file. If no image file is specified, the image file that generated the image currently being viewed is used.
DELGOF	Deletes a record from a graphics overlay file. -REC allows deletion by record number or attribute, and -CURSOR allows deletion by cursor location.
DSTAT	Lists display status. Lists detailed information and status of a specific display, or lists summary information and status of all displays.
ENGRAVE	Allows the user to engrave the displayed graphics data into the displayed image. (Not currently implemented.)
FILL	Replaces the values inside or outside a predefined polygon with the value specified (-VAL) or with values based on the mapping created from the specified break points (-MAP).
FITLIN	Fits a line to a set of points and saves it in the graphics overlay file along with a label and attributes. A minimum of 2 and a maximum of 32 points may be chosen to do the line fit.
FLICKR	Allows the user to flicker several images on the display or to flicker the displayed image through several mappings. FLICKR has two subcommands, FLICKR-IMAGE and FLICKR-MAP. FLICKR-IMAGE will display from two to ten images, one at a time. FLICKR-MAP will display one image through different mappings. From two to ten mappings can be applied to the currently displayed image, one at a time.
FRMDSP	Copies the displayed image from the display's memory plane(s) to a LAS disk file (subcommand -DISK) or into other memory plane(s) (subcommand -MEMORY). Copying the image to disk allows the user to apply the mappings to the image data and/or save the parameters of the image to the associated display parameter file (DPF) and/or save the displayed graphics to the associated graphics overlay file.

HISTO	Displays a histogram for any one or all components of either the currently displayed image (subcommand -IMAGE) or the pixels that comprise a line within the currently displayed image (subcommand -LINE). HISTO generates a histogram of the image currently being displayed either from the "original" data (as it is stored in the display memory planes) or from the data after it has been "mapped."
INIT	Performs a partial or complete initialization of the allocated display using the subcommands -DISPLAY, -IMAGE, -PLANE, -MAP, or -ARL. -DISPLAY does a complete initialization of the display. This includes the image planes, graphics planes, mappings, all DMS tables and all of a user's active record lists. -IMAGE deletes a specified image from the display parameter table. -PLANE initializes the specified image and/or graphics planes. -MAP initializes a mapping to a linear mapping. -ARL deletes any or all of a user's active record lists.
LODDPF	Reads the look-up table (LUT) data from the specified entry in the image's associated display parameter file (DPF) and applies the mappings to the components of the displayed image specified by the parameter MAPCOMP. The user may also choose to apply the zoom and pan factors from the entry to the displayed image.
LODGOF	Generates an active record list of the requested records from a graphics overlay file (GOF). Only records from this list may be drawn and/or manipulated on the display.
LOGIC	Allows the user to perform several logical operations on images. The user may perform a logical AND, OR, or exclusive OR on images stored in the display's memory planes. Note that there must be a memory plane available for storing the resulting image data. All logical operations involve two and only two images.
LSTDPF	Either prints out all of the fields of one specific display parameter file (DPF) entry, or prints out the names of all the entries in the DPF associated with a LAS image. If no LAS image name is entered, the LAS image that generated the image currently being viewed is used. If a DPF entry is specified, then all of the fields of that one specific DPF entry will be printed out. However, if no DPF name is entered, all of the entries in the DPF associated with the image will be printed out. The output for this program can be routed to the terminal, to the printer, or to a text file by the PRINT parameter.

LSTGOF	List out graphics record information to the terminal, line printer, or text file. The graphics records or attribute definitions of the GOF for the displayed image or that of a user-specified image may be listed. In the case of graphics records, the user may specify to list out only the active records.
LSTIMG	Reads the display parameter table (DPT) and prints to the terminal a listing of information for either one or all of the images currently in the memory planes of the allocated display. The information can be displayed in short or long form. The short form includes information about the image name, the protection state of the memories used by the image, the image age, and the memory planes used by each image. The number of free memory planes and a list of the active graphics planes are also displayed to the terminal. The long form includes all of the information in the short form as well as information about the file window from which the image was taken, the bands of the LAS image that were used, the memory window in which the image was placed, the LAS image name from which the image was taken, and the active record list name associated with the image.
MAPP	Allows the user to create and apply mappings to a displayed image. Using the following subcommands, MAPP applies each of the following mappings to the displayed image. -PWL creates a piecewise linear mapping from a set of mapping pairs. -CURSOR creates an arbitrary piecewise linear mapping using the cursor to select breakpoints. -EXP creates an an exponential mapping and scaling. -LOG creates a logarithmic mapping and scaling.
MEASUR	Allows the user to find the ground coordinates of a point location (-POINT), the length of a line (-LINE), or the area of a polygon (-POLY). (Not currently implemented.)
MKMASK	Creates a mask in the graphics plane of specified intensity values from the single-band displayed image.
MODGOF	Allows the user to modify a record in the graphics overlay file. The record may be modified manually or interactively with the cursor.
PIVOT	Allows the user to pivot images. The user may perform a 90, 180, or 270 degree clockwise rotation, a flip (about the horizontal axis), or a mirror (about the vertical axis). Note that for every memory plane used by the input image there must be a memory plane available for storing the resulting image data.

PLANE	Turns graphics plane(s) on in a specified color or turns graphics plane(s) off. The -ON subcommand allows the user to turn graphics plane(s) on in specified color(s). The -OFF subcommand allows the user to turn graphics plane(s) off.
PROTEC	Changes the protection state of the memory planes that an image uses.
PSD	Allows the user to create pseudocolor mappings to be applied to black and white images. PSD-MAN assigns a specified color to a gray level range and/or to specific gray level value(s). PSD-DEF assigns a defined color to a range of image value(s) and/or to specific image value(s). PSD-PIECE defines a pseudocolor mapping through the specification of break points. PSD-PALET allows the user to assign a defined color to a gray level range and/or specific gray level value(s) from a palette using the cursor for color selection (this subcommand has not been implemented).
PUT	Allows the user to interactively place points, lines, polygons and/or annotation into a graphics plane and also create a record in the point, line, polygon or annotation graphics overlay file. The -POINT subcommand allows the user to place points in a graphics plane. The -LINE subcommand allows the user to place a set of points which defines a line in a graphics plane. The -POLY subcommand allows the user to place a set of points which describes the vertices of polygons in a graphics plane. The -ANNOT subcommand allows the user to place annotation in a graphics plane. The graphics plane number can be changed by changing the global variable DMSGPLN or by incrementing the graphics plane number when in pointing device or manual mode.
ROAM	Allows a user to view (in higher resolution) portions of the last image displayed with TODSP. ROAM-WIND will view a user-specified or default window of the last image displayed with TODSP. ROAM-MOVE will view a user-specified neighboring high-resolution window of the last "roamed" image. Neither ROAM-WIND or ROAM-MOVE can view a window that lies outside of the image initially displayed by TODSP.
SAVDPF	Saves an entry from the display parameter table (DPT) in an associated display parameter file (DPF). All parameters necessary to define the specified image are saved in the image's associated DPF.
SAVIMG	Makes a new entry in the display parameter table (DPT) containing the information necessary to redisplay the viewed image in its current state.

SHOGOF	SHOGOF displays and/or erases graphics data from the specified active record list. The user may select (by attribute) data to be displayed and/or erased in the graphics plane indicated by the global parameter DMSGPLN. Graphics displayed with the image use the -IMG subcommand. Graphics may displayed without an image if the user runs the -NOIMG subcommand and specified the window of the graphics to be displayed.
SHOIMG	Displays either an image whose attributes have been saved in the display parameter table (DPT) or an image whose red, green, and blue components are from different entries of the display parameter table. The image data is assumed to be present in the display's memory planes. The bands of the image will be displayed with the mappings and other attributes found in the display parameter table. The user may also choose to associate an active record list name with the displayed image.
SHOMAP	Allows the user to display the values of the mapping applied to the displayed image. The mapping may be written to the terminal, line printer, or to a text file as an input gray level value followed by its red, green, and blue output values or on the display's graphics plane as a graphic representation. Function button 1 may then be used to toggle between displaying a graph of the mapping, a gray level wedge, or both.
SLICE	Sets a range of pixels in the displayed image to a user-specified color. The image must be a single-band black and white image. The user may choose a band slice, a high-level slice, or a low-level slice through the use of the function button.
TIC	Allows the user to place major and minor tic marks in or around an image. The user specifies the number of pixels between each major and minor tic mark.
TODSP	This utility copies single or multiple images or image windows from a LAS image file to a display's refresh memories. TODSP-LOCAL will load the image data from the host system and TODSP-NET will allow the user to load image data from a remote system using the network software to retrieve the data.
ZOOPAN	Allows a user to expand any portion of the image currently being displayed by using the function buttons on the pointing device and to pan over the image using trackball/joystick on the pointing device. The image is zoomed about the center point of the display. The graphics are also zoomed and panned along with the image.

LAS Display Module Overview (LDM)

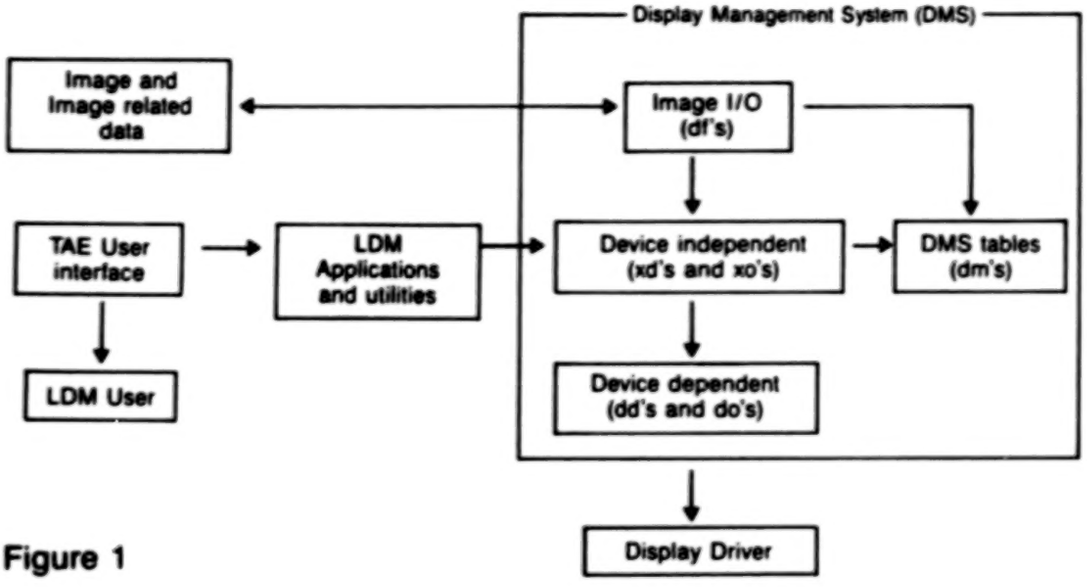


Figure 1

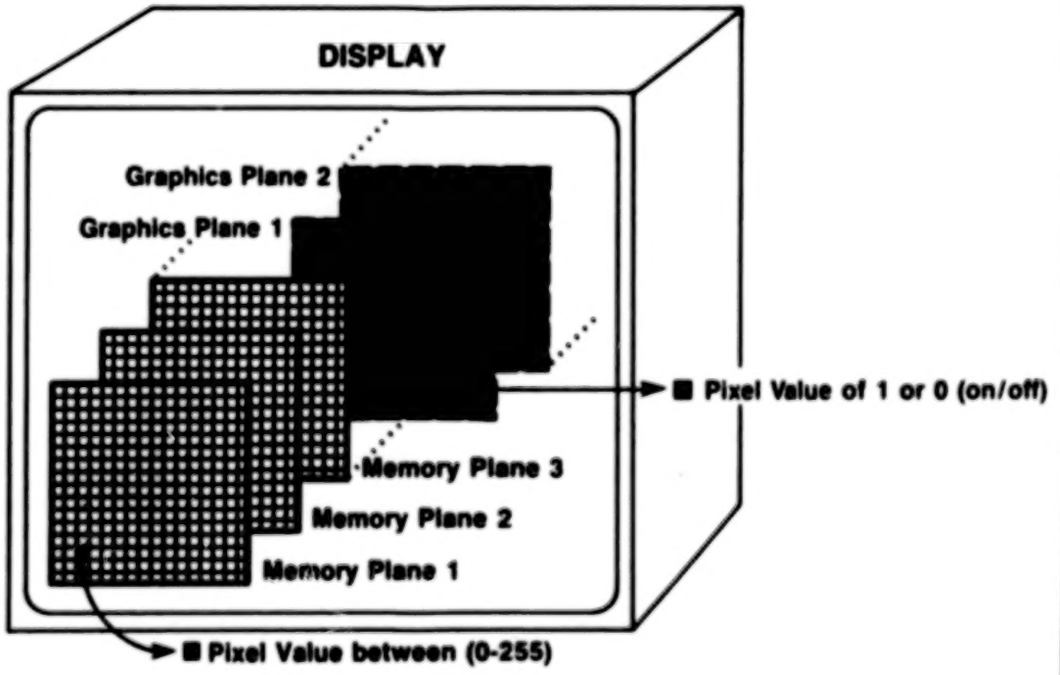


Figure 2

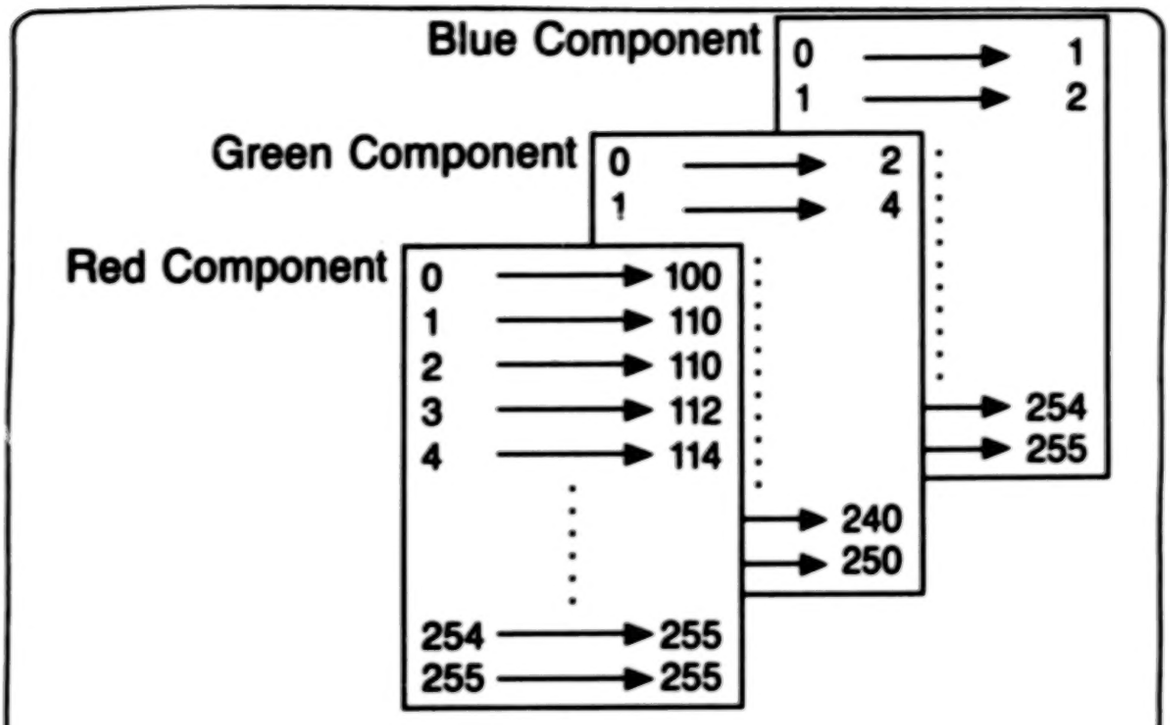


Figure 3

Modifications to DMS tables

Display Device Table (DDT) — Maintains device specific information on each accessible display device.

Active cursor number	→ Added
Cursor shape	→ Added
Cursor size (height and width)	→ Added
Cursor color	→ Added
Cursor blink rate	→ Added
Graphics plane mask	→ Added

Figure 4

Modifications to DMS tables

Display Memory Table (DMT) — Contains information about each refresh memory of the display device.

Memory window	→ To ICT
Source file name	→ To ICT
Shift amount	→ To ICT
Zoom Amount	→ To ICT
Wrap flag	→ To ICT

Figure 5

Modifications to DMS tables

Image Configuration Table (ICT) — Contains information about each image the user has defined
or
Display Parameter Table (DPT)

Memory window	→ From DMT
Source file name	→ From DMT
Shift amount	→ From DMT
Zoom amount	→ From DMT
Wrap flag	→ From DMT
Image group information	→ Removed
File window and bands	→ Added
Look up table data	→ Added
Active record list name	→ Added

Figure 6

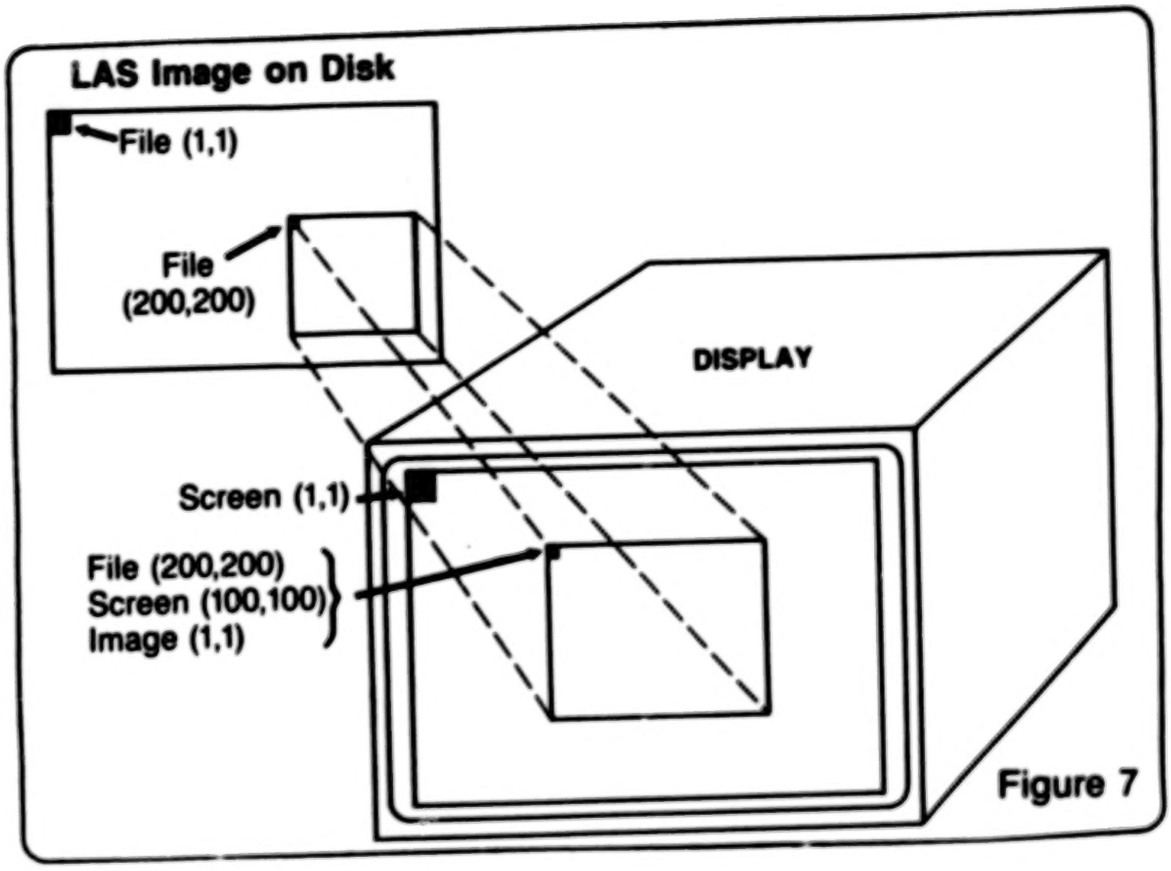

LAS Image on Disk
File (1,1)
File
(200,200)
DISPLAY
Screen (1,1)
File (200,200)
Screen (100,100)
Image (1,1)

Figure 7

LIST OF CONFERENCE ATTENDEES

PRECEDING PAGE BLANK NOT FILMED

SIXTH ANNUAL TAE USERS' CONFERENCE
October 8-10, 1986
Pasadena, California

ATTENDEE LIST

ABBOTT, Mark R. (619) 534-4791
Scripps Institute of Oceanography MABBOTT/NASA
M/S A-002, UCSD
La Jolla, CA 92093

ADAMS, Steven L. (818) 354-2624
Jet Propulsion Laboratory (FTS) 792-2624
4800 Oak Grove Drive, M/S 168-522
Pasadena, CA 91109

ANDERSON, Don (602) 965-6336
Arizona State University D.Anderson/J.P.L.
Geology Department
Tempe, AZ 85287

ASRAR, Ghasem (913) 532-5731
Kansas State University
E.T. Laboratory/Waters Annex Bldg.
Manhattan, KS 66506

BABB, Larry R. (415) 969-6626
Computer Sciences Corporation
444 Castro Street, Suite 400]
Mountain View, CA 94041

BAMBERY,Raymond J. (818) 354-2887
Jet Propulsion Laboratory (FTS) 792-2887
4800 Oak Grove Drive, M/S 168-427
Pasadena, CA 91109

BENNETT, Bart (213) 393-0411
The Rand Corporation
1700 Main Street
Santa Monica, CA 90406

BOIES, Stephen (914) 765-1900
IBM Corporation
Armonk, NY 10504

BOSWOOD, Rebecca Gallegos (213) 594-1540
E-Systems, Melpar Division
Rockwell International
19332 Peachtree Circle
Huntington Beach, CA 92648

PRECEDING PAGE BLANK NOT FILMED

BOTHWELL, Graham (818) 354-3237
Jet Propulsion Laboratory (FTS) 792-3237
4800 Oak Grove Drive, M/S 168-427 GBOTHEWELL
Pasadena, CA 91109

BRUNJES, Kenneth R. (202) 373-3473
Defense Intelligence Agency
DB-5D DIAC RM A4-405
Washington, DC 20340-6644

BURKHART, Brenda (818) 354-0995
Jet Propulsion Laboratory (FTS) 792-0995
4800 Oak Grove Drive, M/S 125-161
Pasadena, CA 91109

CHALFIN, Norman L. (818) 354-2240
Caltech/Jet Propulsion Laboratory (FTS) 792-2240
4800 Oak Grove Drive, M/S 180-302
Pasadena, CA 91109

CHOU, Lang (408) 646-3110
Naval Postgraduate School
63 CQ, NPS
Monterey, CA 93943

COOPER, James (301) 454-5186
University of Maryland
Department of Meteorology
College Park, MD 20742

DeMARTINO, Oscar (818) 354-1295
Jet Propulsion Laboratory (FTS) 792-1295
4800 Oak Grove Drive, M/S 301-260A
Pasadena, CA 91109

ELIASON, Patricia (303) 492-5130
University of Colorado
CIRES, Box 449
Boulder, CO 80309

ENGELBERG, Norman (301) 953-3330
Century Computing, Inc.
1100 West Street
Laurel, MD 20707

FAHY, (James) Benjamin (206) 545-1845
University of Washington
Department of Electrical Engineering, FT-10
Seattle, WA 98195

FANG, Ai C. (202) 453-1693
NASA HQ AFANG
600 Independence Ave., SW, Code EI
Washington, DC 20546

GEMOETS, Richard S. (408) 662-2473
Appaloosa Systems
345 Flume Road
Aptos, CA 95003

GOUGH, Mike (301) 286-6034
Science Applications Research
4400 Forbes Blvd.
Lanham, MD 20706

GUBEREK, Michael (619) 481-5750
Global Imaging, Inc.
201 Lomas Santa Fe Drive
Solana Beach, CA 92075

Guinness, Ed (314) 889-5493
Washington University
Number One Brookings Drive
Earth & Planetary Sciences, Wilson Hall 108
St. Louis, MO 63130

HARRIS, Elfrieda (301) 286-7877
Science Applications Research [TAESO/GSFCMAIL]GSFC
4400 Forbes Blvd.
Lanham, MD 20706

HUFFMAN, George (301) 454-5186
University of Maryland
Department of Meteorology
College Park, MD 20742

HUSSEY, Kevin J. (818) 354-4016
Jet Propulsion Laboratory (FTS) 792-4016
4800 Oak Grove Drive, M/S 168/522
Pasadena, CA 91109

IRANI, Frederick M. (301) 794-5200
Science Applications Research
4400 Forbes Avenue
Lanham, MD 20706

JAMES, Nathan (301) 286-9789
NASA/Goddard Space Flight Center
Greenbelt, MD 20771

JAMIESON, Robert S. (818) 354-5161
Caltech/Jet Propulsion Laboratory (FTS) 792-5161
4800 Oak Grove Drive, M/S 180-302
Pasadena, CA 91109

JEPSEN, Paul L. (818) 354-6047
Jet Propulsion Laboratory (FTS) 792-6047
4800 Oak Grove Drive, M/S 168-427
Pasadena, CA 91109-8099

JOHNSON, Jay A. (301) 731-1006
Bendix Field Engr. Corp.(GSFC)
10210 Greenbelt Road, Suite 310
Seabrook, MD 20738

KINSLOW, Scott (213) 825-2434
University of California/Los Angeles
IGPP
Los Angeles, CA 90024

KOLAR, Michael A. (818) 584-2961
Caltech/Jet Propulsion Laboratory WZHORNE attn. M. KOLAR
M/S 100-22
Pasadena, CA 91125

LENDER, Judy (213) 393-0411
Rand Corporation
1700 Main St., PO Box 2138
Santa Monica, CA 90406

LEE, Steve (213) 825-1555
University of California-Los Angeles
7127 Math Science Building
Los Angeles, CA 90024

LIGGETT, Patricia (818) 354-8456
Jet Propulsion Laboratory (FTS) 792-8456
4800 Oak Grove Drive, M/S 168-522 (818) 354-1405
Pasadena, CA 91109 (FTS) 792-1405

MARTIN, Elizabeth (818) 354-1469
Jet Propulsion Laboratory (FTS) 792-1469
4800 Oak Grove Drive, M/S 301-320 B.MARTIN
Pasadena, CA 91109

MEHLMAN, Robert (213) 825-2434
University of California-Los Angeles RMEHLMAN
IGPP
Los Angeles, CA 90024

MILLER, Philip (301) 953-3330
Century Computing, Inc.
1100 West Street
Laurel, MD 20707

NICHOLS, David A. (818) 354-8912
Jet Propulsion Laboratory (FTS) 792-8912
4800 Oak Grove Drive, M/S 180-703 DNICHOLS/JPL
Pasadena, CA 91109

NICHOLS, J. David (818) 354-4994
Jet Propulsion Laboratory (FTS) 792-4994
4800 Oak Grove Drive, M/S 168-522 JDNICHOLS
Pasadena, CA 91109

PARKER, Dolores (301)286-9514
Science Applications Research
4400 Forbes Boulevard
Lanham, MD 20706

PHILLIPS, Brian (415) 964-9900
Sterling Software
1121 San Antonio Road
Palo Alto, CA 94303

RITTER, Paul R. (415) 642-6346
University of California, Berkeley
Remote Sensing Research Program
260 Space Sciences Lab
Berkeley, CA 94720

SANO, Hajime (818) 354-0995
Jet Propulsion Laboratory (FTS) 792-0995
4800 Oak Grove Drive, M/S 125-161
Pasadena, CA 91109

SESPLAUKIS, Tadas (818) 584-2903
Caltech/JPL WZHORNE, MSG TO: T SESPLAUKIS
M/S 100-22
Pasadena, CA 91125

SHERES, David (202) 767-5270
Naval Research Laboratory
Code 7782
Washington, DC 20375

SIRRI, Norri (818) 354-5740
Jet Propulsion Laboratory (FTS) 792-5740
4800 Oak Grove Drive, M/S 168-427
Pasadena, CA 91109

SLAVNEY, Susan (314) 889-5493
Washington University
Number One Brooking Drive
Earth & Planetary Sciences, Wilson Hall 108
St. Louis, MO 63130

SMITH, Gary (818) 584-2946
Federal Electric Corp., ITT Service
Caltech/IPAC, M/S 100-22
Pasadena, CA 91125

STANFILL, Daniel (818) 354-3742
Jet Propulsion Laboratory (FTS) 792-3742
4800 Oak Grove Drive, M/S 168-427
Pasadena, CA 91109

STEPHENSON, Gerald D. (601) 688-5264
NORDA
Building 1105, Code 321
Natl Space Tech Lab, MS 39529

SULLIVAN, Scott (818) 354-1416
Jet Propulsion Laboratory/Telos Corp. (FTS) 792-1416
4800 Oak Grove Drive, M/S 301-350A
Pasadena, CA 91109

SYNOTT, Debra J. (818) 354-4512
Jet Propulsion Laboratory (FTS) 792-4512
4800 Oak Grove Drive, M/S 301-375H
Pasadena, CA 91109

SZCZUR, Martha (301) 344-1176
Goddard Space Flight Center
Code 521
Greenbelt, MD 20814

THOMPSON, Donald M. (202) 373-3514
Defense Intelligence Agency
DB-5D, DIAC Bolling Air Base
Washington, DC 20301

TORSON, James (602) 527-7254
US Geological Survey JTORSON
2255 N. Gemini Drive
Flagstaff, AZ 86001

WALKER, Richard (818) 354-3534
Jet Propulsion Laboratory (FTS) 792-3534
4800 Oak Grove Drive, MIPL-M/S 168-522
Pasadena, CA 91109

WITTMAN, Dennis K.
Caltech/JPL
IPAC, M/S 100-22
Pasadena, CA 91125
(818) 584-2969
WZHORNE, MSG TO D. WITTMAN

WOLF, Karl
Century Computing, Inc.
1100 West Street
Laurel, MD 20707
(301) 953-3330

ZION, Philip
Jet Propulsion Laboratory
4800 Oak Grove Drive, M/S 169-236
Pasadena, CA 91109
(818) 354-8730
(FTS) 792-8730

ZOBRIST, Al
The Rand Corporation
1700 Main Street, PO Box 2138
Santa Monica, CA 90406
(213) 393-0411
x 6480

BIBLIOGRAPHIC DATA SHEET

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA CP-2463		

4. Title and Subtitle	5. Report Date
TAE Sixth Annual Users' Conference	October 1986
	6. Performing Organization Code: 521
7. Author(s): Martha Szczur, Elfrieda Harris*, Editors	8. Performing Organization Report No.: 87B0176
9. Performing Organization Name and Address: Goddard Space Flight Center, Greenbelt, Maryland 20771	10. Work Unit No.
	11. Contract or Grant No.
	13. Type of Report and Period Covered: Conference Publication
12. Sponsoring Agency Name and Address: National Aeronautics and Space Administration, Washington, D.C. 20546	14. Sponsoring Agency Code

15. Supplementary Notes

*Science Applications Research; Lanham, Maryland 20706

16. Abstract

The purpose of the conference was to allow users to interchange information on the use of TAE and its various applications as well as facilitate interaction between users and developers on the future of TAE. To meet these intentions, this year's conference offered an excellent balance of time for presentations, demonstrations, special interest discussions and informal discussions.

The presentations and conversations demonstrated that many TAE users have the same needs and goals. There are several TAE sites interested in developing device-independent imaging applications, accessing data bases, designing User Interface Management Systems (UIMS), executing TAE on PC's, creating programming tools and utilizing windowing systems. It is hoped that this meeting will result in solid work toward coordination between TAE sites and future enhancements to TAE.

17. Key Words (Selected by Author(s))	18. Distribution Statement
TAE, UIMS, Prototyping, Executive	Unclassified-Unlimited Star Category 61

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified		

*For sale by the National Technical Information Service, Springfield, Virginia 22161

GSFC 25-44 (10/77)

END

DATE

FILMED

AUG 5 1987